(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 January 2002 (17.01.2002)

PCT

(10) International Publication Number
WO 02/05104 A1

(51) International Patent Classification⁷: G06F 15/16, 17/00

(21) International Application Number: PCT/US01/21187

(22) International Filing Date: 3 July 2001 (03.07.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/216,822	7 July 2000 (07.07.2000)	US
Not furnished	2 July 2001 (02.07.2001)	US

(71) Applicant: **INTERACTUAL TECHNOLOGIES, INC.** [US/US]; 100 Century Center Court #205, San Jose, CA 95112 (US).

(72) Inventors: **LAMKIN, Allan, B.**; 4282 Farley Lane, San Diego, CA 92122 (US). **COLLART, Todd, R.**; 206 Arbuelo Way, Los Altos, CA 94022 (US).

(74) Agent: **SAMPLES, Kenneth, H.**; Fitch, Even, Tabin & Flannery, Room 1600, 120 South LaSalle Street, Chicago, IL 60603 (US).

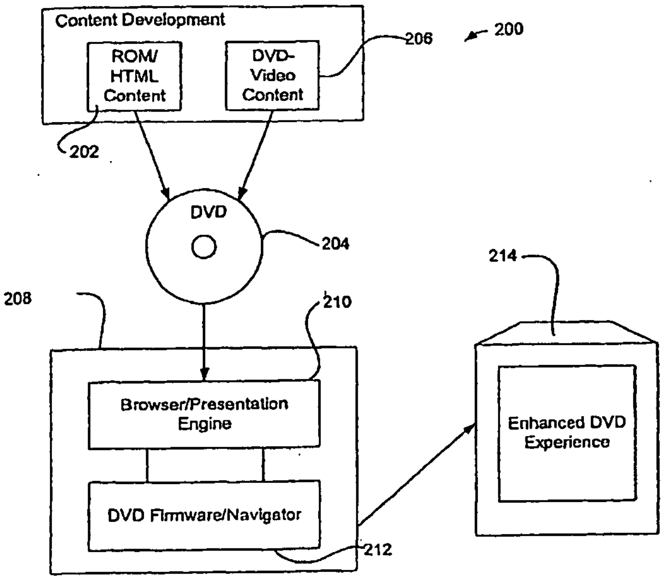
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

[Continued on next page]

(54) Title: SYSTEM, METHOD AND ARTICLE OF MANUFACTURE FOR A COMMON CROSS PLATFORM FRAMEWORK FOR DEVELOPMENT OF DVD-VIDEO CONTENT INTEGRATED WITH ROM CONTENT



(57) **Abstract:** A method for providing enhanced content (214) for play across multiple play platforms employs steps of delivering media content (206) to a client device; delivering HTML content (202) to a client device, the HTML content being accessible and usable by a plurality of client device platforms; activating a browser (210) to access the HTML content, the browser being located on and compatible for use with the client device; activating firmware (212) on the client device to access the media content; and incorporating the accessed HTML content with the accessed media content.

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SYSTEM, METHOD AND ARTICLE OF MANUFACTURE FOR A COMMON CROSS PLATFORM FRAMEWORK FOR DEVELOPMENT OF DVD-VIDEO CONTENT INTEGRATED WITH ROM CONTENT

This patent document claims priority to Provisional Patent Application No. 60/216,822 under 35 U.S.C. § 119(e).

BACKGROUND OF THE INVENTION

The present invention relates to enhancement of multimedia content and more particularly to a system, method and apparatus for enhancing multimedia content with supplemental content.

Multimedia computer systems have become increasingly popular over the last several years due to their versatility and their interactive presentation style. A multimedia computer system can be defined as a computer system having a combination of video and audio outputs for presentation of audio-visual displays. A modern multimedia computer system typically includes one or more storage devices such as an optical drive, a CD-ROM, DVD (DVD-Video or DVD Audio etc), Laser Disc, Video Disc or Audio Disc, or a hard drive. Audio and video data are typically stored on one or more of these mass storage devices. In some file formats the audio and video are interleaved together in a single file, while in other formats the audio and video data are stored in different files, many times on different storage media. Audio and video data for a multimedia display may also be stored in separate computer systems that are networked together. In this instance, the computer system presenting the multimedia display would receive a portion of the necessary data from the other computer system via the network cabling.

Multimedia computer systems have become increasingly popular over the last several years due to their versatility and their interactive presentation style. A multimedia computer system can be defined as a computer system having a combination of video and audio outputs for presentation of audio-visual displays. A modern multimedia computer system typically includes one or more storage devices such as an optical drive, a CD-ROM, a hard drive, a videodisc, or an audio disc, and audio and video data are typically stored on one or more of these mass storage devices. In some file formats the audio and video are interleaved together in a single file, while in other formats the audio and video data are stored in different files, many times on different storage media. Audio and video data for a multimedia display may also be stored in separate computer systems that are networked together. In this instance, the computer system presenting the multimedia display would receive a portion of the necessary data from the other computer system via the network cabling.

Graphic images used in Windows, multimedia applications can be created in either of two ways, these being bit-mapped images and vector-based images. Bit-mapped images comprise a plurality of picture elements (pixels) and are created by assigning a color to each pixel inside the image boundary. Most bit-mapped color images require one byte per pixel for storage, so large bit-mapped images create correspondingly large files. For example, a full-screen, 256-color image in 640-by-480-pixel VGA mode requires 307,200 bytes of storage, if the data is not compressed. Vector-based images are created by defining the end points (corners), thickness, color, pattern and curvature of lines and solid objects within an image. Thus, a vector-based image includes a definition that consists o umerical representation of the coordinates of the object, referenced to a corner of the image.

Bit-mapped images are the most prevalent type of image storage format, and the most common bit-mapped-image file formats are as follows. A file format referred to as BMP is used for Windows bit-map files in 1-, 2-, 4-, 8-, and 24-bit color depths. BMP files contain a bit-map header that defines the size of the image, the number of color planes, the type of compression used (if any), and the palette used. The Windows DIB (device-independent bit-map) format is a variant of the BMP format that includes a color table defining the RGB (red green blue) values of the colors used. Other types of bit-map formats include the TIF (tagged image format file), the PCX (Zsoft Personal Computer Paintbrush Bitmap) file format, the GIF (graphics interchange file) format, and the TGA (Texas Instruments Graphic Architecture) file format.

The standard Windows format for bit-mapped images is a 256-color device-independent bit map (DIB) with a BMP (the Windows bit-mapped file format) or sometimes a DIB extension. The standard Windows format for vector-based images is referred to as WMF (Windows meta file).

Full-motion video implies that video images shown on the computer's screen simulate those of a television set with identical (30 frames-per-second) frame rates, and that these images are accompanied by high-quality stereo sound. A large amount of storage is required for high-resolution color images, not to mention a full-motion video sequence. For example, a single frame of NTSC video at 640-by-400-pixel resolution with 16-bit color requires 512K of data per frame. At 30 flames per second, over 15 Megabytes of data storage are required for each seconc full motion video. Due to the large amount of storage required for full motion video, various types of video compression algorithms are used to reduce the amount of necessary storage. Video compression can be performed either in real-time, i.e., on the fly during video capture, or on the stored video file after the video data has been captured and stored on the media. In addition, different video compression methods exist for still graphic images and for full-motion video.

Examples of video data compression for still graphic images are RLE (run-length encoding) and JPEG (Joint Photographic Experts Group) compression. RLE is the standard compression method for Windows BMP and DIB files. The RLE compression method operates by testing for duplicated pixels in a single line of the bit map and stores the number of consecutive duplicate pixels rather than the data for the pixel itself. JPEG compression is a group of related standards that provide either lossless (no image quality degradation) or lossy (imperceptible to severe degradation) compression types. Although JPEG compression was designed for the compression of still images rather than video, several manufacturers supply JPEG compression adapter cards for motion video applications.

In contrast to compression algorithms for still images, most video compression algorithms are designed to compress full motion video. Video compression algorithms for motion video generally use a concept referred to as interframe compression, which involves storing only the differences between successive frames in the data file. Interframe compression begins by digitizing the entire image of a key frame. Successive frames are compared with the key frame, and only the differences between the digitized data from the key frame and from the successiv mes are stored.

Periodically, such as when new scenes are displayed, new key frames are digitized and stored, and subsequent comparisons begin from this new reference point. It is noted that interframe compression ratios are content-dependent, i.e., if the video clip being compressed includes many abrupt scene transitions from one image to another, the compression is less efficient. Examples of video compression which use an interframe compression technique are MPEG, DVI and Indeo, among others.

MPEG (Moving Pictures Experts Group) compression is a set of methods for compression and decompression of full motion video images that uses the interframe compression technique described above. The MPEG standard requires that sound be recorded simultaneously with the video data, and the video and audio data are interleaved in a single file to attempt to maintain the video and audio synchronized during playback. The audio data is typically compressed as well, and the MPEG standard specifies an audio compression method referred to as ADPCM (Adaptive Differential Pulse Code Modulation) for audio data.

A standard referred to as Digital Video Interactive (DVI) format developed by Intel Corporation is a compression and storage format for full-motion video and high-fidelity audio data. The DVI standard uses interframe compression techniques similar to that of the MPEG standard and uses ADPCM compression for audio data. The compression method used in DVI is referred to as RTV 2.0 (real time video), and this compression method is incorporated into Intel's AVK (audio/video kernel) software for its DVI product line. IBM has adopted DVI as the standard for displaying video for its Ultimedia product line. The DVI file format is based on the Intel i750 chipset and is su ed through the Media Control

Interface (MCI) for Windows. Microsoft and Intel jointly announced the creation of the DV MCI (digital video media control interface) command set for Windows 3.1 in 1992.

The Microsoft Audio Video Interleaved (AVI) format is a special compressed file structure format designed to enable video images and synchronized sound stored on CD-ROMs to be played on PCs with standard VGA displays and audio adapter cards. The AVI compression method uses an interframe method, i.e., the differences between successive frames are stored in a manner similar to the compression methods used in DVI and MPEG. The AVI format uses symmetrical software compression-decompression techniques, i.e., both compression and decompression are performed in real time. Thus AVI files can be created by recording video images and sound in AVI format from a VCR or television broadcast in real time, if enough free hard disk space is available.

As discussed above, such audio and video content is often stored on media such as CD-ROM or digital video disc (DVD). However, once a vendor has delivered such content to a customer, the vendor loses any practical control over the product. Even if the product is delivered under license rather than out right sale, it has traditionally been difficult to prevent a customer from copying the content or providing the content to any number of friends so that they might illegally copy the content.

The now familiar compact disk preserves information as a series of microscopic pits and smooth areas, oriented in concentric circular or helical tracks, on the otherwise smooth, planar surface of an annular disk. Recorded information is read from a c :t disk by directing a focused laser beam along the recorded tracks, and detecting variations in the intensity of the laser beam along the recorded tracks, and detecting variations in the intensity of the laser beam as it encounters the microscopic pits and smooth areas on the disk. The coherence and relatively short wavelength of laser radiation enables large volumes of information to be written onto very small spaces of a recording medium.

Compact disks were first introduced in the music recording industry in 1982, and now account for 43% of all recorded music sales. In the United States alone, over three hundred million compact disks are sold annually, with a retail value of over three billion dollars, according to the Recording Industry Association of America. The most prevalent format for recording multimedia events onto such disks is Digital Video or Versatile Disk (DVD). The DVD is a read only format for recording a relatively large amount of high quality data. When delivered to a user, the disk is input into a CD-ROM player on a client device such as a computer. Software on the client device allows the DVD formatted data to be read.

Once the DVD disk has been manufactured the content is essentially fixed. The content that the user can access from the disk is limited to the content provided when the disk was manufactured. In order to update the information, a new disk must be created and delivered to the user. This is an expensive and inconvenient solution.

Thus there remains a need for a system for easily and efficiently updating content provided on a DVD-disk. Such a system would preferably allow update information to be delivered via a network such as the Internet. In addition, such a system would take advantage of software capabilities already present on the clien ice, and would importantly be able to function on the many different possible platforms of client devices, such as for example Macintosh, PC or a set top box.

Disc technologies that are re-writeable like a CD-RW or technologies that allow multiple sessions can be used for adding additional or updated content directly to the disc. Thus for multi-session discs, where the first session of the disc is write-once and additional sessions on the disc can be either write-one, or rewriteable, additional or updated content can be added to these additional sessions of the disc. This includes such technologies as the "Orange Book" specification for CD-ROM, including CD-PROM and Multimedia discs such a Dataplay.

Flash memory based and other similar memory technologies can be used for storing multimedia and additional or updated content as well. This includes IBM technology that uses a USB interface to coupled a personal computer to a storage device such as a "keychain" memory device.

The present invention advantageously addresses the above and other needs.

SUMMARY OF THE INVENTION

The present invention advantageously addresses the needs above as well as other needs by providing the enhancement of multimedia content and more particularly to providing a system, method and apparatus for enhancing multimedia content with supplemental content.

In one embodiment, the invention can be characterized as a method for providing enhanced content for play across multiple play platforms. The method employs steps of delivering media content to a client device; delivering

HTML content to a client device, the HTML content being accessible and usable by a plurality of client device platforms; activating a browser to access the HTML content, the browser being located on and compatible for use with the client device; activating firmware on the client device to access the media content; and incorporating the accessed HTML content with the accessed media content.

In another embodiment, the invention can be characterized as a method for enhancing multimedia content. The method employs steps of providing a recording medium; recording content onto the recording medium; integrating HTML content with the recorded content; accessing the recorded content and the HTML content; and playing a multimedia event based on the accessed content.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and other aspects, features and advantages of the present invention will be more apparent from the following more particular description thereof, presented in conjunction with the following drawings wherein:

Fig. **1** is a schematic diagram of a hardware implementation of one embodiment of the present invention;

Fig. **2** is a schematic diagram of a system for enhancing a DVD multimedia experience;

Fig. **2A** is a flow chart illustrating steps traversed upon insertion of a DVD disk (or other media) into a device, such as a DVD player;

Fig. **3** is a flowchart illustrating logic for incorporating update information to supplement a DVD multimedia play experience;

Fig. **4** is graphical representation of data layouts for bitmap layers;

Fig. **5** is a flowchart illustrating a method for providing an enhanced multimedia experience; and

Fig. 6 is a flowchart illustrating a method for enhancing DVD content with ROM content.

Corresponding reference characters indicate corresponding components throughout the several views of the drawings.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The following description of the presently contemplated best mode of practicing the invention is not to be taken in a limiting sense, but is made merely for the purpose of describing the general principles of the invention. The scope of the invention should be determined with reference to the claims.

Fig. 1 illustrates a system for providing enhanced DVD content for play across multiple user platforms. Both DVD-Video content and HTML content are recorded on DVD discs and provided to a user. The HTML content includes various directories that allow it to be accessed by multiple platforms of user devices. Once inserted into a user device, browser software on the user device accesses the HTML content and supplies supplemental update information to enhance the play experience provided by the DVD-Video content. The supplemental update information can be either retrieved via a network such as the Internet or can be provided directly from the HTML data itself stored on the DVD disc.

In various embodiments, the client devices may take the form of computers, televisions, stereos, home appliances, or any other types of devices. In one embodiment, the client apparatuses and the host computer each include a computer such as an IBM compatible computer, Apple Macintosh computer or UNIX based workstation.

A representative hardware environment is depicted in

Fig. **1**, which illustrates a typical hardware configuration of a workstation in accordance with a preferred embodiment having a central processing unit **110**, such as a microprocessor, and a number of other units interconnected via a system bus **112**. The workstation shown in Fig. **1** includes a Random Access Memory (RAM) **114**, Read Only Memory (ROM) **116**, an I/O adapter **118** for connecting peripheral devices such as disk storage units **120** (i.e. DVD playback device) to the bus **112**, a user interface adapter **122** for connecting a keyboard **124**, a mouse **126**, a speaker **128**, a microphone **132**, and/or other user interface devices such as a touch screen (not shown) to the bus **112**, communication adapter **134** for connecting the workstation to a communication network (e.g., a data processing network) and a display adapter **136** for connecting the bus **112** to a display device **138**. The workstation typically has resident thereon an operating system such as the Microsoft Windows NT/2000 or Windows 95/98/ME Operating System (OS), the IBM OS/2 operating system, the MAC OS, or UNIX operating system. Those skilled in the art will appreciate that the present invention may also be implemented on platforms and operating systems other than those mentioned.

A preferred embodiment is written using JAVA, C, HTML and the C++ language and utilizes object oriented programming methodology. Object oriented programming (OOP) has become increasingly used to develop complex applications. As OOP moves toward the mainstream of software design and development, various software solutions require adaptation to make use of the benefits of OOP. A need exists for these principles of OOP to be applied to a messaging interface of an electronic messaging system such that a set of OOP classes and objects for the messaging interface can be provided.

A preferred embodiment of the invention utilizes

Hypertext Markup Language (HTML) to implement documents on the Internet together with a general-purpose secure communication protocol for a transport medium between the client and the Newco. HTTP or other protocols could be readily substituted for HTML without undue experimentation. Information on these products is available in T. Berners-Lee, D. Connoly, "RFC 1866: Hypertext Markup Language - 2.0" (Nov. 1995); and R. Fielding, H, Frystyk, T. Berners-Lee, J. Gettys and J.C. Mogul, "Hypertext Transfer Protocol -- HTTP/1.1: HTTP Working Group Internet Draft" (May 2, 1996). HTML is a simple data format used to create hypertext documents that are portable from one platform to another. HTML documents are SGML documents with generic semantics that are appropriate for representing information from a wide range of domains. HTML has been in use by the World-Wide Web global information initiative since 1990. HTML is an application of ISO Standard 8879; 1986 Information Processing Text and Office Systems; Standard Generalized Markup Language (SGML).

To date, Web development tools have been limited in their ability to create dynamic Web applications which span from client to server and interoperate with existing computing resources. Until recently, HTML has been the dominant technology used in development of Web-based solutions. However, HTML has proven to be inadequate in the following areas:

- Poor performance;
- Restricted user interface capabilities;
- Can only produce static Web pages;
- Lack of interoperability with existing applications and data; and
- Inability to scale.

Sun Microsystem's Java language solves many of the client-side problems by:

- Improving performance on the client side;
- Enabling the creation of dynamic, real-time Web applications; and
- Providing the ability to create a wide variety of user interface components.

With Java, developers can create robust User Interface (UI) components. Custom "widgets" (e.g., real-time stock tickers, animated icons, etc.) can be created, and client-side performance is improved. Unlike HTML, Java supports the notion of client-side validation, offloading appropriate processing onto the client for improved performance. Dynamic, real-time Web pages can be created. Using the above-mentioned custom UI components, dynamic Web pages can also be created.

Sun's Java language has emerged as an industry-recognized language for "programming the Internet." Sun defines Java as: "a simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high-performance, multithreaded, dynamic, buzzword-compliant, general-purpose programming language. Java supports programming for the Internet in the form of platform-independent Java applets." Java applets are small, specialized applications that comply with Sun's Java Application Programming Interface (API) allowing developers to add "interactive content" to Web documents (e.g., simple animations, page adornments, basic games, etc.). Applets execute within a Java-compatible browser (e.g., Netscape Navigator) by copying code from the server to client. From a language standpoint, Java's core feature set is based on C++. Sun's Java literature states that Java is basically, "C++

with extensions from Objective C for more dynamic method resolution."

Another technology that provides similar function to JAVA is provided by Microsoft and ActiveX Technologies, to give developers and Web designers wherewithal to build dynamic content for the Internet and personal computers. ActiveX includes tools for developing animation, 3-D virtual reality, video and other multimedia content. The tools use Internet standards, work on multiple platforms, and are being supported by over 100 companies. The group's building blocks are called ActiveX Controls, small, fast components that enable developers to embed parts of software in hypertext markup language (HTML) pages. ActiveX Controls work with a variety of programming languages including Microsoft Visual C++, Borland Delphi, Microsoft Visual Basic programming system and, in the future, Microsoft's development tool for Java, code named "Jakarta." ActiveX Technologies also includes ActiveX Server Framework, allowing developers to create server applications. One of ordinary skill in the art readily recognizes that ActiveX could be substituted for JAVA without undue experimentation to practice the invention.

In accordance with one embodiment, a cross-platform DVD specification defined, which is called InterActual Technologies Cross Platform, hereafter referred to by the name ITX. By following the ITX specification, DVD authors can create HTML-enhanced DVD-Video/Audio content that can play reliably across multiple playback platforms, ranging from computers (such as Windows and Macintosh) to Internet-connected set-top devices (such as the Sony Playstation II and Nuon-enhanced consumer DVD players). The general requirements for enhanced DVD authoring and the requirements for the playback devices, both hardware and software are described herein.

The ITX enables DVD-Video/Audio (hereafter referred to only as DVD-Video) content developers to create products that seamlessly combine the Internet and/or other DVD-ROM capabilities with DVD-Video to create a richer, more interactive, and personalized entertainment experience for their customers. All this is accomplished without the need for content developers to create special content for each unique playback platform, and without the need of becoming an expert programmer on Windows, Macintosh, and other platforms. Additionally the present invention allows for customized content and functions tailored for specific platform(s).

Internet connectivity is not a requirement for the use of ITX. A stand-alone system with HTML browser functionality is all that is required. In addition, CD-DA (standard music CDs) can also be enhanced by use of ITX.

The following terms are defined as follows:

Term	**Description**
BCA	Burst Cutting Area. Area near inner ring on a DVD disc where custom data can be imprinted
ATVEF	Advanced Television Enhancement Forum (spec allows combining HTML and TV programming)
PIP	Picture in Picture
DVD-Video	A disc authored in accordance with the DVD-Video specification. Any place where the term DVD-Video is used it also applies to DVD-Audio,

	unless specifically excluded.
UOP	User Operations (as defined DVD-Video and DVD-Audio specifications)

The following documents are incorporated by reference:

1.	HTML Cross Platform Authoring Guidelines
2.	ISO-9660
3.	ATVEF Specification (http://www.atvef.com)
4.	DVD-Video (Book 3) and DVD-Audio (Book 4) specifications

The ITX specification provides a common framework whereby content developers, browser providers, and hardware manufacturers can successfully create and playback Internet-enhanced DVD and CD products.

This description of the embodiments is divided into three major sections, targeting three different audiences:

- ***Content Development Requirements***: Addresses issues specific to DVD authors and content creators. The target audience includes DVD authoring facilities, web designers, and graphics and creative production facilities. This section outlines the integration of DVD-Video with Web Pages, Programming Interfacing, and other cross-platform DVD-Video and DVD-ROM authoring considerations.

- ***Browser Requirements***: Addresses issues specific to browser implementation. The target audience includes establishments such as: PlanetWeb, Spyglass, Liberate, and VM Labs (with a custom implementation of the Spyglass browser). This section outlines basic browser requirements to support ITX titles and integration of a DVD-Video programming interface.

- ***Platform/Hardware Requirements***: Addresses issues specific to DVD-Video hardware platforms. The target audience includes specialized chip manufacturers, consumer DVD-Video player, game system manufacturers (Sony Playstation, Nintendo, Sega), and any others who might incorporate web connectivity into DVD player products. This section outlines display requirements, browser interfaces, and other hardware-specific requirements.

With reference to Fig. **2**, in an embodiment, a system 200 is provided for enhancing an internet play experience. ROM/HTIM content 202 is recorded onto a DVD disc 204. Additionally DVD-Video content 206 is also recorded onto the DVD disc. The disc 104 is inserted into a client device 208 that contains Browser/Presentation software 210 thereon. The client device hardware also includes a DVD Firmware/Navigator 212 that reads the DVD-Video content. In addition, the client device 208 includes a Browser/Presentation Engine software 110, which reads the ROM/HTML Content. The Browser/Presentation Engine can be for example Netscape Navigator or some other engine commonly available on personal computers. After reading the ROM/HTML content, the browser software 110 searches the Internet to find supplemental information related to the DVD content and incorporates the supplemental information into the DVD content 202, 206 to create an Internet Enhanced DVD Experience 214.

To better understand the purpose and goals of ITX three possible usage scenarios are described, each with an increasing level of complexity.

DVD-Video disc with movie script provided:

A movie is authored with the entire screenplay provided on the DVD disc in HTML format. Clicking on any scene visually represented in the HTML immediately links the user to that scene within the DVD-Video. Besides being a finer granularity than the normal chapter navigation provided on DVD-Video, the HTML-based script could contain other media (pictures, audio) and/or live web links for other information (stored either on the DVD disk, or accessible through the Internet). Further, the text of the screenplay in HTML could automatically "scroll" with the DVD-Video to give the appearance of being synchronized with the DVD-Video. Although many of these types of features (minus live web links and synchronized scrolling) could be authored in DVD-Video, HTML authoring is much more efficient, immediate and widely known.

More complex menus:

A DVD-Video is shipped with a simple HTML page that does little except start a movie. However, the HTML page also uses the Internet and checks to see if that movie has any web site updates. If it does, then the HTML page launches a new movie menu that is downloaded from the web. This new menu might have e-commerce opportunities (buy gifts based on the movie; buy tickets for the sequel to the DVD, etc.). Because the new movie menu is not on the DVD-Video, but rather is on a server accessible via the internet, the window of time during which the choices on the new movie menu is available can be decided by the studio long after the DVD-Video has shipped. The new movie menu may have new links to an actor's web site, which can be particularly advantageous if, for example, the actor has become a star since the movie was made, and therefore wasn't given star treatment in the original DVD-Video. The new movie menu may just be a more convenient way to navigate the disc to a finer granularity than the chapters provided. Advantageously, in accordance with the present embodiment, the DVD can have new movie menus stored on a server accessible through the internet, and that can be changed over time. If the DVD-Video is played without ITX, the DVD-Video operates in a conventional manner.

Live webcast with the director or stars in live chat:

A DVD-Video movie is shipped with an HTML page that links the user to an ITX web site. This site (and studio advertising) notifies the user of the date/time of a "live chat" with, for example, the movie's director, who will discuss the making of the movie. Near the event starting time, the user connects to the web site with the DVD-Video in his/her DVD player. At the start time, the director begins sending voice (such as streaming audio, e.g., Real Audio) over the Internet. The director controls the DVD player of the user, as well as other DVD players, by sending play, pause, fast-forward, and rewind commands, etc. (Latecomers are automatically synchronized). User (if they have, for example, a browser with a keyboard) can enter questions. The director can choose which questions to answer and control every DVD player to an appropriate scene in the movie and discuss the scene. Through the use of bitmap overlay layers and drawing tools, the director can pause the video and draw on the screen (like a football play) to better explain the details involved in creating a certain scene, for example. As the director moves from one question to another the video can use transitions and special effects to make the presentation more professional and entertaining.

In order to support the above-described functionality, the present embodiment is as follows. An ITX disk can contain DVD Video and ITX-compatible ROM data, DVD-Audio and ITX-compatible ROM data, CD-Audio and ITX-compatible ROM data, or the like. The ITX compatible ROM data can be any digital file type including HTML and graphics, including for example, HTML graphics, subject to file system limitations described below. There is no theoretical limit to the amount of ITX compatible ROM data that can be placed on a DVD disk, except for physical constraints of the DVD disk (or in an alternative embodiment CD-disk).

An ITX-compatible disk adheres to rules regarding capability, detection, file system, directory structure, and content location, each of which is described in further detail herein. As multi-platform support is a goal of the present embodiment, an ITX disk provides for both platform-specific behavior and general-purpose behavior. Platform-specific behavior can be accomplished using the ITX-API described above, by either placing platform-specific binaries on the DVD disk in predefined directories, described herein, or by authoring general purpose HTML content that uses, for example, ECMAScript or JavaScript and the ITX API to detect specific platforms and to "serve" web pages specifically designed for a particular platform, i.e., particular type of device. General purpose content can be created for playback on multiple platforms using HTML content and the ITX API. Both approaches can be combined so that platform-specific behavior can be employed with certain devices, while other devices, such as devices developed after release of the DVD disk, can be supported with general-purpose behavior.

For personal computers, such as personal computers operating under Microsoft Windows, ITX content can be viewed through a proprietary browser client per the ITX content viewed through the proprietary browser client can be the same content that is displayed on, for example, a browser-enhanced consumer electronics system, such as a set top box, a game console, or an internet-connected DVD player or the like.

The device must provide a capability to determine the type of media that has been inserted into the device. Specifically, the device must be able to determine whether the media is a DVD disk or some form of CD disk. For CD-DA, there may or may not be a file structure formatted on the CD-DA disk, such as described herein and therefore the CD-DA disk table of contents must be read per the "red-book" specification.

An ITX-compatible DVD or CD is detected by checking for the existence of a file named index.htm in a directory named common. The ITX-API version information can be found in a mediated area in the index.htm file, which is an HTML file.

The index.htm file provides JavaScript that detects the particular type of device into which the DVD disk has been inserted, and the device is "navigator," and provides general framework for interactive playback. For a disk not authored in accordance with the ITX content, a content homepage is employed, i.e., file named default.htm is employed. The default.htm file may be stored in memory on the device.

CONTENT DEVELOPMENT REQUIREMENTS:

This section describes the requirements for content developers, DVD authors, and creative houses. This section outlines how DVD-Video can be integrated with ROM content for playback across multiple hardware platforms and multiple browsers. For more information regarding cross-platform HTML development (independent of integration with DVD-Video), refer to the InterActual™ HTML Cross Platform Authoring Guidelines document. This reference document outlines platform/browser detection, use of JavaScript files (.js files) and other HTML authoring techniques.

DVD/ROM Authoring Considerations:

ITX Directory and File Naming Conventions (mandatory compliance)

When making an ITX disk, DVD video zone files must be placed physically at the beginning of the ITX disk, contiguously, in the order specified by the DVD-Video specification, likewise, DVD-Audio zone files must follow the DVD-Video files in contiguous order.

The DVD specifications for DVD-Video and DVD-Audio require that each disc contain specific directories and files. For example, the DVD-Video files are contained in a directory (or folder) with the name VIDEO_TS; DVD-Audio files in the AUDIO_TS directory. The VIDEO_TS and AUDIO_TS directories should be the first entries in the directory descriptor (the true order of the directory and file entries is usually hidden, since most operating systems list them in, for example, alphabetical order). There is no such requirement for "DVD-ROM" content, and, thus, developers can arrange other files on a disc in any desired manner. It is best to place ROM-zone files in subdirectories versus the root directory.

The placement of files on a dual-layered disk (DVD-9, DVD-14, or DVD-18) is generally independent of layer details. DVD-Video and DVD-Audio files must begin on layer zero. ROM-zone files are beginning after the DVD-Video (or DVD-Audio) files and can cross layer boundaries, if needed. In order to prevent problems that can arise from this open aspect of the specifications, ITX provides a convention for ordering and naming files.

Files stored for use with ITX can be in any DVD disc directory. However, there must be a method that allows the platform-specific browser and/or playback engine to identify the initial starting HTML file in the case were there is no executable file. Also, in order to simplify support, it is strongly suggested that the full convention described below be followed on all ITX-authored discs.

ITX Naming Standard:

Each DVD-Video authoring system and tool set supports different naming capabilities; such as ISO-9660, ISO-9660 with Joliet extensions, Macintosh file names, support for Macintosh resources, hybrid discs, etc. Some authoring tools go even further by forcing a certain character case (e.g., the Toshiba authoring system forces all characters to uppercase). These issues must be taken into account as part of the development process since some playback platforms may operate differently depending on the physical layout and file structure on the DVD. As a specific example, Windows and Macintosh operating systems are case *insensitive*, whereas Unix and Linux operating systems are case *sensitive*.

For ITX compliance, the following naming standard must be followed:

- UDF 1.02 and ECMA 167 (second edition)
- Support for hybrid Windows/Macintosh discs (whereby resource forks for the Macintosh operating system are preserved)

All files and directories must be developed with case *sensitivity* in mind. The recommended approach is to use only capital letters for all directories, file names, and HTML references. To be safe, only use A-Z, 0-9 and the underscore. The initial HTML file shall have a name of ITX.HTM.

ITX Directories:

The ITX.HTM file must be located in a directory that follows these rules. Other files, based on individual authoring needs may be located in any directory following any convention. There may be more than one ITX.HTM file. For example, there could be a different one for each platform supported, or just one primary one and one alternate for a single platform that requires special operations.

Directory name	**Platform**
COMMON	All (default)
LINUX	linux operating system
MAC	Macintosh
NINTENDO	Nintendo Dolphin
SONY	Playstation II and CE
NUON	VMLabs
WIN	Windows
SEGA	Sega Dreamcast

TOSHIBA	Toshiba
WIN	Windows
ZORAN	Zoran
To be determined	all other ITX directory names must be registered to insure no conflicts.

Directory Naming Conventions

Note that any new platform directory names should be reserved and assigned before use. However, each platform developer can control the directory structure under its reserved top-level directory name. For example, Sony could create a PS2 and PS3 directory under the SONY directory.

This directory structure allows for proprietary executable binary files for each platform. For example, a current PCFriendly DVD (i.e., a DVD in accordance with the present embodiment) can utilize the directory structure by placing the Windows version of software in a WIN directory, and a Macintosh version of software in a MAC directory. Upon insertion of the disc, the platform will initiate execution of the appropriate binaries (based on some platform-specific autorun feature) and then the binaries will load the ITX.HTM file.

The set-top player browser shall locate its starting file via the following logic:

- Check for online updates enabled and online
- If OK, then check web for update and use, if found
- Else check for its platform-specific directory.
- If the platform-specific directory exists and the ITX.HTM

file is found, then start;

- Else if the COMMON ROM directory exists and the ITX.HTM file is found, then start;
- Else the disc is not an ITX disc and it should be played as a DVD-Video disc.

The above-described structure allows for device specific executable binary file for each type of device supported by a particular DVD disk. The platform-specific directory structure and its associated set of binaries enable any platform to run executables specifically designed for any device provided that such executables are available on the particular DVD disk being utilized. This capability, in essence, allows the device specific binaries to override general purpose ITX content or override a standard browser mechanism. While the actual ROM content may reside in a device specific directory, it is recommended that all content reside in the common directory when possible. The common directory can support any number of subdirectories, including device specific subdirectories.

The common directory stores, in most cases, the actual ITX content (versus platform specific binaries). It is recommended that all ITX content (even platform specific ITX content reside in the common directory as this provides an intuitive content development approach. By maintaining a single content directory, Java Script can easily be used to detect platforms and render appropriate HTML players pages tailored to specific devices.

There may be cases where device specific binaries may be included on the DVD disk, but still the general-purpose content. For example, an ITX disk can utilize the directory structure by placing a Windows version of software in the WIN directory, and the Macintosh version of the software in the MAC directory. Upon insertion of the ITX disk, the platform will initiate execution of the appropriate binaries (based on a device specific feature, such as autorun) and then the binaries will load the index.htm file located in the common directory, the starting point for any general-purpose ITX disk.

The starting or entry point is the index.htm file, with which resides in the top level of the common directory. It is recommended that all ITX content (with the exception of device-specific binaries) be stored in the common directory. Java Script can then be used to detect platforms and render appropriate HTML pages tailored to specific platforms. The index.htm file will be the background "container" web page while standard playback occurs. This page enables Java Script event handlers to be loaded and activate to handle events during playback. The meta-data of the index.htm file contains the ITX-API version information.

Referring to Fig. **2A**, a flow chart is shown of steps traversed upon the coupling of a storage medium with, e.g., insertion of a DVD disk into, a device, e.g., a personal computer, a consumer electronics device, a game console, or the like. Steps traversed upon insertion of the DVD disk into the device are divided into phases as follows. During the first phase, a disk detection algorithm determines if the disk has content in accordance with the present embodiment, i.e., whether the disk is an ITX disk, i.e., whether the disk contains ITX content. This determination is made by examining whether a file named index.htm is located in a "common" directory on the DVD disk.

If the index.htm file exists, then the DVD disk is an ITX disk. Otherwise, the DVD disk is determined not to be an ITX disk. During a second phase, a determination is made as to whether the DVD disk is a DVD-Video or a DVD-Audio, or whether a disk of another type has been inserted, such as a CD-DA. Logic for performing the second phase is included, generally, in the device, and this is not discussed in detail further herein. (Such logic is known.)

During a third phase, a determination is made as to a default playback mode of the device. This is determined by reading a "player mode" from the property, e.g., InterActual.PlayerMode. If the device is configured for "play" mode, ITX content, e.g., HTML content, is bypassed, whereas if the device is configured for ITX mode, then the ITX content is launched beginning with the index.htm file in the common directory. The ITX content itself can then be updated dynamically if the device is connected to the Internet, or an equivalent network. There is no Internet connection, or equivalent connection, the device renders ITX content from a ROM portion of the DVD disk.

For non-ITX disks, when the device is configured for ITX mode, a default content home page (called default.htm) is displayed and an Internet connection, or equivalent, is attempted to provide potential ITX content for the non-ITX disk.

During a fourth phase, platform specific file detection occurs, and a determination is made as to whether there are platform specific binary files on the DVD disk for the device. This is accomplished by searching for a predefined directory, as described above, associated with the device.

During a fifth phase, a determination is made as to whether a connection to the internet or similar network, can be made. This step is performed for ITX disks in order to determine whether updated content is available from a server. Additionally, for DVD disks without ITX content, a connection to an on-line database can be attempted, so that the database can be interrogated to determine whether a server containing content associated with the DVD disk is available. If such content is available, an interactive experience similar to that available on ITX disk can be offered to the user of the device. When the device is in "play" mode, then ITX disk can display an icon, to signify that ITX content is available from the DVD disks ROM area. If the user selects the icon, a content home page is displayed, i.e., default.htm, so that the user can switch to ITX mode.

With reference to Fig. **3**, a process **300** for obtaining update information is described. The process **300** begins with a decision step **302** wherein a determination is made as to whether the user is online and whether the user prefers to check for updates online. If the answer to decision step **302** is no, then in another decision step **304** a determination is made as to whether HTML update information is available. If such information is available then in an operation **306**, the ITX.HTM is started from the web and the update information is retrieved. If the answer to question **302** or both questions **302** and **304** are no, then in yet another decision step **308** a determination is made whether a platform directory exists, the platform directory applicable to the platform of the user device. If an appropriate platform directory does exist, then in an operation **310**, ITX.HTM is started in the platform directory. If an appropriate platform directory does not exist, then in a decision step **312** a determination is made as to whether a common directory exists which can be used with the platform of the user device. If such a common directory does exist, then ITX.HTM is started in that directory. If such a common platform directory does not exist, then in a step **316** the DVD is played a normal video without Internet enhancement.

It is recommended that each player have a user setup that allows the ITX functionality to be overridden, such as:

- Check for ITX and start as ITX if found (default setting)
- Check for ITX and give the user a menu choice of ITX or Standard
- Show the ITX icon on the screen for several seconds when ITX is found (include a remote control function that re-starts discs in ITX mode
- Play all discs as DVD-Video, ignoring ITX

ITX Programming Interface (mandatory compliance):

This section describes the ITX application programming interface (API) for controlling and scripting ITX-enhanced discs. The API is divided into five sections:

- **Embedding.** Syntax for embedding DVD-Video within a web page. This section also addresses displaying video full screen and in a window.
- **Commands.** Commands control the playback and search mechanisms of a DVD-Video disc.
- **Properties.** Properties are used to query attributes of the DVD-Video and set certain configuration properties.
- **Events.** Events are used to trigger notification of various playback conditions, such as time changes, title changes and UOP changes. Events are essential for

scripting and synchronizing the video with other assets.

Embedding:

This section describes how to embed DVD-Video within an HTML page and control its layout.

Computer operating systems shall embed DVD-Video using currently available embedding techniques. Examples for each of the major computer operating systems is are provided below:

Operating System	**Example**

Windows	<object ID="PCFriendly"
	CLASSID="clsid:A0739DE5-571F-11D2-A031-0060977F760C"
	BORDER="1" WIDTH=50% HEIGHT=60% >
	</object>

Apple/Macintosh	<embed ID=”PCFriendly”
	TYPE=”application/x-pcfriendly-plugin”
	ALT=”PCFriendly Plug In”
	HIDDEN=”TRUE” >
	</embed>

Linux	TBD
Others	TBD

Examples for embedding DVD-Video in HTML

After the DVD-Video object is embedded in the web page, it can be accessed using any style sheet, link, or scripting language. Values for the ID string must begin with a letter (A-Z or a-z) and may be followed by any number of letters, digits, hyphens, and periods up to a maximum of 48.

Unlike computers, set-top boxes do not generally have a full-blown operating system and browser. Therefore, the capabilities within the browser are often more restricted. For embedding DVD-Video within these platforms using ITX, the "PCFriendly" ID must be integrated within the embedded browser as any other tag structure. With this approach, any embedded browser that encounters the "PCFriendly" tag, would automatically associate this identifier with the ITX programming API described later in this section.

While many possible windows configurations are possible for displaying the update data, the update data is preferably provided on a screen in the following manner:

- Toggling between full screen
- Displaying within frame
- Dynamic resizing

Commands:

Commands (also known as functions or *methods* in OOP terminology) control the playback and navigation mechanisms of a DVD-Video/Audio or CD-DA disc. Commands can be used by the calling application (HTML, C++, or other) to initiate a DVD/CD playback function. The commands supported by ITX are listed below.

- All commands support return values as shown above.

- See notes at the end of the table and for a description of the optional time and FX parameters designated by [*].

- Items in square brackets [] are optional.

ITX Commands	Description	DVD Player	CD-DA	Support Level	Range
Open(filename | type)	Opens specified file name. See note 1.	Both	Y	Base	-
Play([*])	Start playback of the DVD.	Both	Y	Base	-
Pause([*])	Pause playback of the DVD (Pause_On). Subsequent issue of Pause() resumes playback (Pause_Off)	Both	Y	Base	-
Stop([*])	Stop playback of the current DVD. Stops execution of current PGC and transfers to Stop State.	Both	Y	Base	-
FastForward([x[,*]])	Fast forward the current DVD at x speed. (By default, x = 2)	Both		Base	2 - 99, although some players may allow only the DVD values of

					2,4,8,16,32
Rewind([x[,*]])	Rewind the current DVD at x speed. (By default, x = 2)	Both		Base	same as above
Slow([x[,*]])	Play the current DVD at x speed. (By default, x = 2 for ½ speed). Supported Features should be checked first to determine if capability is supported. See note 2.	Both		Adv	2 - 99 (reciprocal values)
SlowReverse(x[,*])	Play the current DVD at x speed in reverse. (By default, x = 2 for ½ speed). Supported Features should be checked first to determine if capability is supported. See note 2.	Both		Adv	same as above
Step([n[,*]])	Steps playback of the DVD forward n frames. Supported Features should be checked first to determine if	Both		Adv	1 - 30

	capability is supported. By default, n=1.				
NextChapter([*])	Halts playback of the current chapter and starts playback from the next chapter within the title.	Video		Base	-
PrevChapter([*])	Halts playback of the current chapter and starts playback from the start of the current chapter; or if near the start of a chapter goes to the previous chapter.	Video		Base	-
NextTrack()	Halts playback of the current track and starts playback from the next track in the same Audio Title within the Title Group.	Audio	Y	Base	-
PrevTrack()	Halts playback of the current track and starts playback from the previous track in the same Audio	Audio	Y	Base	-

	Title within the Title Group.				
NextDisplay([*])	Presents the next visual display/slide in the display list to the user.	Audio		Base	-
PrevDisplay([*])	Presents the previous visual display/ slide in the display list to the user.	Audio		Base	-
TitlePlay(t[,*])	Start playback at the specified title index, t. (Initializes all GPRMs and SPRMs).	Video		Base	1 - 99
ChapterPlay(t,c[,*])	Start playback at the specified title index and chapter value. If in TT_DOM and already within specified title, ChapterSearch is issued to maintain all GPRM and SPRM values. Otherwise, ChapterPlay is issued and all GPRMs and SPRMs are initialized.	Video		Base	t: 1 - 99 c:1 - 99
TimePlay(h,m	Start playback at	Both		Base	h: 00 -

,s,f[,*])	the specified. Specify time in hours, minutes, seconds, frames. (Computer must translate into milliseconds). If in TT_DOM or TT_GR_DOM and already within specified title, TimeSearch is issued to maintain all GPRM and SPRM values. Otherwise, TimePlay is issued and all GPRMs and SPRMs are initialized.				23 m: 00 - 59 s: 00 - 59 f: 00 - 29
TitleGroupPlay(g[,*])	Start playback at the specified title group number.	Audio		Base	g: tbd
TrackPlay(g, t[,*])	Start playback at the specified title group number and track number. If in TT_GR_DOM and already within specified title group, TrackSearch is issued to maintain all GPRM and SPRM values.	Audio	Y	Base	g: tbd t: tbd

	Otherwise, TrackPlay is issued and all GPRMs and SPRMs are initialized. In case of CD-DA, group number should be 1 by default.				
HiddenGroupPlay (g[,*])	Plays desired hidden/locked group.	Audio		Adv	g: tbd
HiddenTrackPlay (g,t[,*])	Plays desired hidden/locked track within Hidden Group.	Audio		Adv	tbd tbd
HiddenTimePlay (h,m,s[,*])	Plays from specific time within Hidden Group.	Audio		Adv	h: 00 – 23 m: 00 – 59 s: 00 – 59
Menu(x[,*])	Display the specified menu. See note 3.	Both		Base	1 – 5
GotoMenuID(x[,*])	Displays menu associated with entered menu ID	Both		Adv	–
GotoBookMark (x[,*])	Continues playback at the specified bookmark location by number. See note 4.	Both	Y	Adv	–

SaveBookMark (x[,*])	Creates a bookmark at the current location to store with a given number. See note 4.	Both	Y	Adv	0 - n n is system dependent, about 32
Resume([*])	Resume DVD playback (if applicable based on Navigation).	Both		Base	-
StillOff([*])	Continue with still off.	Both		Base	-
UOPMask()	Retrieve current UOPs.	Video		Base	-
AutoMouseHide(b)	Show or hide the mouse cursor when the DVD is playing. (Hide occurs 2 seconds after no activity)	Both		Adv	
UpButtonSelect([n])	Selects the up direction button n times. By default n = 1.	Both		Base	1 - 36
DownButtonSelect ([n])	Selects the down direction n times. By default n = 1.	Both		Base	1 - 36
LeftButtonSelect([n])	Selects the left direction button n times. By default n = 1.	Both		Base	1 - 36
RightButtonS	Selects the right	Both		Base	1 - 36

elect ([n])	direction button n times. By default n = 1.				
ButtonActivate()	Activate the current highlighted button.	Both		Base	-
ButtonSelect And Activate(n)	Activate the specified highlighted button, where n is the button number between 1 and 36.	Both		Base	1 - 36
AudioSelect(n)	Sets the stream number of the Audio to play	Both		Base	1 - 8
SubPictureSelect(n)	Sets the stream number of the Subpicture to display	Video		Base	1 - 32
SubPictureEnable(n)	Enables or Disables Subpictures (sub titles)	Video		Base	0 = off 1 = on
AngleSelect (n)	Sets the stream number of the Angle to play	Video		Base	1 - 8
MenuLanguage Select (n)	Selects the language for the System Menu according to the language code (n). Only available in	Both		Base	1 - tbd

	a Stop State.				
TextLanguage Select (n)	Selects the language for the Audio Text Data.	Audio		Base	1 - tbd
ParentalLevelSelect(n)	Selects parental level of player.	Video		Base	1 - 8
ParentalCountrySelect(n)	Selects the country for the parental level.	Video		Base	1 - tbd
KaraokeSelect(x)	Changes the Audio mode for Karaoke.	Video		Adv	1: vocalist 1 2: vocalist 2 3: guide melody
Zoom([x,y[,*]])	Zoom (or scale) by a percentage factor of x (horizontal) and y (vertical). Individual players may support various zoom ranges, but 25% to 400% is recommended (2500 < x,y < 40000). See note 5. X and Y are integers, 100	Both		Adv	2500 - 40000 (16-bit unsigned values)

	times the percentage. By default, x and y are 10000 (100%).				
Pan([x,y[,*]])	Set center point of zoomed display to x,y coordinates based on percentage of normal content full screen display. See note 5. X and Y are integers, 100 times the percentage. By default, x and y are zero (center point).	Both		Adv	-5000 to +5000
VideoBlending ([a,c[,*]])	Controls whether the video is played in its own window/full screen (a=0) or if the video is in the background and HTML content is blended on top of it (where a is the alpha blending value from 1 to 255; c is HTML the HTML background	Both		Adv	a: 0 – 255 c: 32-bit ARGB

	color that is clear). By default, a=255 (HTML on top) and c=white (white HTML background is clear for video to show through). See note 6.				
Bitmap Layer Extensions					
CreateLayer(b,c,r,d,p)	Create an overlay layer b is the bitmap overlay ref number (or handle) Initialize to color c r is the resolution d is the number of bits per pixel. p is the palette when b=1,2,4 or 8 See note 7.	Both	Y	Adv	b: 1 - 9 c: 32-bit ARGB r: 1 - 4 d: 1, 2, 4, 8, 15, 16, 24, 32 p: palette tbl
ChangePalette(b,p[,*])	Change the palette for layer b	Both	Y	Adv	b: 1 - 9 p: palette tbl
DestroyLayer (b)	Destroy an overlay layer. If b= 0	Both	Y	Adv	1 - 9

	then destroy **all** layers.				
ShowLayer(b[,*])	Make a layer visible	Both	Y	Adv	1 - 9
HideLayer(b[,*])	Hide a layer	Both	Y	Adv	1 - 9
SetVectorDraw (b,c,w[,*])	Set default drawing color (c) and width (w) for layer b. c: ARGB or index into palette w: 1 to 16 pixels	Both	Y	Adv	b: 1 - 9 c: 32-bit ARGB or palette index w: 1 - 16
SetVectorCorners (x1,y1,x2,y2)	Set the user coordinate system to: x1,y1 (upper left corner); x2,y2 (lower right corner). The default is 0,0..720,480 for NTSC and 0,0..720 576 for PAL; which matches one to one to pixels in the layer. (All values are 16-bit signed integers) This coordinate system is used for VectorMove and				x: 0 - 720 (or 1920) y: 0 - 480, 576 (or 1080)

	VectorDraw				
VectorMove(x,y,b[,*])	Move to x,y on layer b. (x,y based on SetVectorCorners)	Both	Y	Adv	x: 0 – 720 (or 1920) y: 0 – 480, 576 (or 1080) b: 1 – 9
VectorDraw(x,y,b[,*])	Draw to x,y on layer b. (x,y based on SetVectorCorners)	Both	Y	Adv	x: 0 – 720 (or 1920) y: 0 – 480, 576 (or 1080) b: 1 – 9
DisplayImage (f, b, a[,*])	Display image from file (types: JPEG, Gif) in layer b, with alpha blend level a. (Layer must have sufficient bit depth.)	Both	Y	Adv	f: filename b: 1 – 9 a: 0 – 255
Time and FX					
SetRelTime([t])	Set the relative time counter to t. (unsigned 32-bit integer in ms) See GetRelTime	Both	Y	Base	t: 0 – 2^{31} unsigned 32-

	property. The relative time affects only commands issued for delayed execution (queued). Once queued commands use machine absolute time. By default, t = 0.				bit integer
FlushCmdQueue()	Flush and/or initialize the command queue.	Both	Y	Base	
Misc Extensions					
SetMixVolume(x[,*])	Set primary audio stream volume level to percentage x to allow overmixing.	Both	Y	Adv	x: 0 - 100 0 = mute main audio
FullScreen(w[,*])	Toggles video playback between full screen and window.	Both		Base	0 = full 1 = window
NetConnect([u[,*]])	Establish WWW connection to optional URL provided. By default, connection is made	Both	Y	Adv	u = text string

	with no URL.				
Computer Only:	**The following commands should be ignored in set-top players**				-
Close()	Close the driver and stop playback of the current DVD.	Both	Y	Base	–
ShowControls (x,y[,*])	Show or hide the video controls in full screen mode, at x,y. Use -1,-1 to hide.	Both	Y	Base	x: 0 – 1920 y: 0 – 1200
ShowContextMenu ()	Controls the right mouse click context menu.	Both		Base	–
PopUpMenu()	Displays and allows the audio languages, sub-pictures, and angles to be set to those currently available.	Both		Base	–
SuppressErrors(b)	Suppresses display of error messages (0= suppress display of error messages, 1= display error messages)	Both	Y	Base	0 = suppress 1 = show

ITX Command Summary

Command Notes:

[*] optional parameters

Special effects and timed operations are performed with four optional parameters. See section 2.1.2.2.3 for complete details.

1. Open.

Opening of VOB files and MPEG files is required for baseline support. Other file types are advanced features. An open file can be played, paused, stopped. Fast forward and rewind are not available. Stopping causes the file pointer to be reset to the start of the file.

2. Slow and Slow Reverse.

If slow is supported a speed of ½ is required. Other slow speeds may also be supported; decreasing powers of two are recommended: ¼, 1/8, 1/16, etc although any value from ½ to 1/99 is allowed. Integer reciprocal values are used for the speeds, such as 2 for ½ and 4 for ¼, etc.

3. Menu.

Menu choices are:

1: Title Menu
2: Root Menu
3: Chapter Menu
4: Audio Languages Menu
5: Subpicture Languages Menu

4. Bookmarks.

The bookmarks are assigned a number when set. A GotoBookMark returns to the same position on the disc as when the bookmark was set (saved). Preservation of bookmarks during powerdown is not required, however, if implemented, bookmarks shall be unique to the disc (using a generated disc id). A minimum of one bookmark per disc is required if implemented (32 recommended). It is recommended that bookmarks save the entire DVD-Video or DVD-Audio state, but this is not required. At a minimum, the correct title and time must be saved.

5. Zoom and Pan.

Zoom parameters are based on a percentage, so integer values of 10000 and 10000 (x and y) indicate 100% of normal full screen display with no zoom. Normally the x and y scale factors should be the same to maintain a correct aspect ratio. When zooming to a value greater than 100%, by default, the center point of the image remains on the center of the display. Panning allows moving the center point of the portion of the image to be displayed. These x and y pan parameters are provided as a percentage of the display from -50% to +50% using integer values from -5000 to +5000. (This is done so that the differences between NTSC and PAL do not have to be calculated in pixels. Additionally, it may also be possible to use the same HTML code for handling 4:3 and 16:9 as well.) If the pan parameters would cause the display to pan off the edge of the video, then the platform software shall only set that panning parameter to the largest or smallest value that keeps the video in the display area.

6. Blending.

This advanced feature allows an HTML page to be constructed that includes a background color (the colorkey) that is treated as clear. Other information on the page (graphics or text) is then alpha blended with the video. An alpha value of 0 indicates that the video shows everywhere. An alpha value of 255 indicates that the HTML page shows everywhere (except where it is clear as defined by the colorkey value). This allows, for example, placing textual titles on top of the video (or blended with the video). Graphical menus can be added in the same manner. A minimum of 16 (256 recommended) discreet alpha values are required if this feature is supported. However, the alpha blend parameter is always from 0 to 255.

7. Bitmap Layers.

The bitmap layer features allow defining and using multiple layers (possibly only one active at a time depending on the playback device) with other layers stored in memory and ready to be activated when needed. This is how an event moderator can remotely draw onto the video image. The number of bits per pixel (color depth) can be 1, 2, 4, 8, 15, 16, 24 or 32. For bpp of 1, 2, 4 and 8 a palette must be provided (32-bit ARGB data). It is anticipated that the most frequently used capability will be bpp values of 1 or 2 to be used for image markup, like a chalkboard with 1 to 4 colors. Bpp values of 15, 16, 24 and 32 allow images to be used on a layer.

Errors and Warnings

All commands shall return one of the following error codes

number	**name**	**description**	**commands**
0	OK	Successful	all
1	General Error	Other or Unknown error condition	all
1	FileNot Found	File not found	Open, DisplayImage
2	NotSupp	File type or feature not	

	orted	supported	
3	NoDisc	Attempt to play with no disc	Play, + others?
4	BadParam	Parameter out of range	many
5	ParamError	Parameter out of range for current disc or current condition	many
6	NoMem	Not enough memory for operation	CreateLayer, DisplayImage
7	QueueFull	Command queue is full	Time delayed command
8	QueueFail	Timed command error (such as overlapping command)	Time delayed command
9	QueueWarn	Timed command accepted, but action may be emulated	Time delayed command

Error and Warning Summary

Layers and pixel formats:

With reference to Fig. **4**, a data layout **400** for a layer with 2 bpp, resolution is depicted. Also depicted is a data layout **402** for a layer with a 4bpp resolution. The pixel formats for data in a layer may be stored (internally) in any format. The diagrams on the next page illustrate how it might work in one implementation. The data in the LUT (lookup table) is provided as a color palette by CreateLayer() or ChangePalette() . Each entry in the palette is a 32-bit integer as follows:

byte 0: blue (least significant byte)
byte 1: green

byte 2: red
byte 3: alpha

An alpha value of 0 indicates a transparent color (i.e. the video shows through) and 255 is a solid opaque color (i.e. no video shows through). The first entry in every color palette should consist of 4 bytes of zero, a clear color. The default color value, c, in CreateLayer() should normally be zero to initialize the layer to a clear color. Special visual effects can be created by use of other values, such as initializing a layer to red, then erasing it off with a series of drawing commands and/or by changing alpha values, in the color palette, etc. Layers and data on them are not affected by video transitions and special effects having to do with video playback.

As described in the next section, some commands may be modified with time parameters and special effects or transitions. Of particular note to the bitmap layer commands are the ChangePalette() and VectorDraw() commands. A VectorDraw() command with a time duration simply draws a line incrementally. However, the ChangePalette() command with a time duration should be implemented such that there are three complete palettes, the original, the final and the current palette. At each time increment every palette table entry is interpolated towards its final value. The ShowLayer() and HideLayer() commands may have a timed special effect applied to them such as a wipe or fade.

Transitions, Special Effects and Timing:

Similar to how an author might use transitions and special effects during video editing, ITX allows a subset of these types of capabilities, depending on the unique capabilities of each playback system. If a system cannot produce the effect due to hardware or software limitations then it should gracefully degrade to some emulation or simply produce no effect at all, but concluding at the same logical end point.

Transitions can be used, for example, when switching from one scene to another with a time search or chapter search. If no effect is specified, then the playback system would normally produce a standard cut or possibly insert black frames between the scenes. However, if a wipe left is specified, then the final still frame of scene 1 is shown and scene 2 wipes in from the left at the specified rate. (No attempt is made to provide a moving image for both scenes simultaneously.)

The following table details the optional parameters and their ranges:

	Description	**range**
xxx(?[t1,t2,fx,p]) see command list in 2.1.2.2 where [*] is replaced by [t1,t2,fx,p]	Command xxx with optional parameters as follows: t1: time 1 (0 or relative start time) t2: time 2 (duration (if t1=0) or relative end time) fx: special effect number p: extra parameter based on fx There are two basic modes: 1. If t1=0, then t2 is the command duration. 2. If t1>0, then t1=starting	t1: 0 - 2^{31} t2: 0 - 2^{31} fx: 0 - 999 p: 32-bit value based on x all unsigned

	relative time and t2=ending relative time. t1 and t2 are unsigned 32-bit integers in milliseconds. The possible choices for the variable numbers of parameters are: t1 (not allowed) t1, t2 t1, t2, fx t1, t2, fx, p	32-bit integers

Optional Parameter format for timed commands and special effects

Notes:

1. Immediate Execution

   To cause a timed command or special effect to start immediately, the t1 parameter must be set to zero (or to a value less than the current relative time). The t2 parameter contains the duration of the command (when t1 = 0). If t1 is greater than zero, but less than the current relative time then the duration is equal to t2-t1. A negative duration is treated as the shortest possible time for that operation.

2. Delayed Execution

   To cause a command to be queued for later execution, the t1 parameter must be set to a non-zero value greater than the current relative time. To accomplish this the current relative time can be queried via the

GetRelTime property. Alternatively, the relative time can be set using the SetRelTime command. Once a command has been queued, the player shall convert the relative time to an absolute time for its scheduled execution and cannot be changed. (However, the command queue can be flushed.)

3. SpecialFX.

   Any immediate or delayed execution command can have a special effect or transition (with optioannl parameter) added to modify its operation. All special effects and transitions must be accepted by all players but may be emulated or ignored if the effect cannot be performed. The same is true of the timed nature of various commands - if a player has a fixed duration for executing a particular command, then the requested duration is ignored.

4. Command Macros

   Macros of commands can be created by using the SetRelTime and then issuing various commands based with offsets from that time.

5. Command queue

   The player must support a command queue with a depth of at least two items (eight is recommended; PC/Mac: 64 is recommended). That is, two items are pending execution at a later time while further commands continue to execute. If a command is accepted for the queue, then it must be executed (unless flushed or some other operation negates or overrides its action). Times stored in the queue should be in an absolute machine time (not relative time and not DVD playback time) so that subsequent changes to the relative time do not affect commands already queued.

6. Conflicting commands

   Because it is possible to schedule commands that have overlapping times, these must be checked prior to acceptance for the queue. Non-conflicting, overlapping operation can be accepted. Conflicting overlapping operations may be accepted also if the operations can still be logically completed. Conflicting overlapping operations that are accepted shall return a warning code. An example of a conflicting operation would be to schedule a chapter advance with a 5 second fade in and a second chapter advance after only 2 seconds. Robust internally interlocks must be used if there is any chance of an erroneous program to lock up a machine due to the use of timed or delayed execution commands. A fallback to basic sequential operation is suggested.

Exemplary transitions and special effects according to an embodiment of the invention include the following.

num	name	description	parameters
0	none	standard cut, no effect	none
1	dissolve	old scene dissolves away, new scene appears	none
2	fade	old fades to color, new scene fades in	color: 32-bit ARGB
3	wipe	old scene is wiped off revealing new scene	LRTB
4	reveal	old scene is pulled off, revealing new scene	LRTB
5	slide	new scene slides on, covering old scene	LRTB
6	push	new scene pushes old scene	LRTB

		off	
7	peal	old scene is peeled off (like a wipe, but 3D)	LFTB
8	corner wipe	wipe from a corner	ULURLLLR
9	corner reveal	reveal from a corner	ULURLLLR
10	corner slide	slide from a corner	ULURLLLR
11	corner peal	peel from a corner	ULURLLLR
12	random boxes	random boxes poke holes in old scene revealing new scene	box size in pixels
13	blinds	horiz or vert blinds wipe off revealing new scene	blind size in pixels
14 – 99		reserved for other transitions	
100	none	standard video and audio	none
101	YUV	adjustments are made to luma and chroma	byte 0: V byte 1: U byte 2: Y byte 3: reserved 0 (signed byte adjustments)
102	snow	snow is added to the display	0 = none 255 = maximum
103	ripple	video is played like underwater	0 = none 255 = maximum
10		reserved	

4-9999			
1000 and up		assignable for specific system effects	

List of Transitions and Special Effects

Notes:

LRTB: 1 = left, 2 = right, 3 = top, 4 = bottom

ULURLLLR: 1 = upper left, 2 = upper right, 3 = lower left, 4 = lower right

All transitions and/or effects do not make sense with each command. The guiding philosophy should be to implement only those that make sense. The following table is the *recommended* set features for the most advanced playback systems with Y_1 being the most basic to Y_4 the most advanced.

ITX Command	**time delayed (queued)**	**time duration**	**effect/transition**
ChangePalette (b,p[,*])	Y_1	Y_4	
ChapterPlay(t,c[,*])	Y_1	Y_2	Y_2
DisplayImage(f, b, a[,*])	Y_1	Y_3	Y_3
FastForward([x[,*]])	Y_1	Y_2	Y_2

FullScreen(w[,*])	Y_1	Y_3	
GotoBookMark(x[,*])	Y_1		
GotoMenuID(x[,*])	Y_1	Y_2	Y_2
HiddenGroupPlay(g[,*])	Y_1		
HiddenTimePlay(h,m,s[,*])	Y_1		
HiddenTrackPlay(g,t[,*])	Y_1		
HideLayer(b[,*])	Y_1	Y_3	Y_3
Menu(x[,*])	Y_1	Y_2	Y_2
NetConnect([u[,*]])	Y_2		
NextChapter([*])	Y_1	Y_2	Y_2
NextDisplay([*])	Y_1	Y_2	Y_2
Pan([x,y[,*]])	Y_1	Y_2	
Pause([*])	Y_1	Y_2	Y_2
Play([*])	Y_1	Y_2	Y_2
PrevChapter([*])	Y_1	Y_2	Y_2
PrevDisplay([*])	Y_1	Y_2	Y_2
Resume([*])	Y_1	Y_2	Y_2
Rewind([x[,*]])	Y_1	Y_2	Y_2
SaveBookMark(	Y_1		

x[,*])			
SetMixVolume(x[,*])	Y_1		
SetVectorDraw(b,c,w[,*])	Y_1		
ShowControls(x,y[,*])	Y_1	Y_3	Y_3
ShowLayer(b[,*])	Y_1	Y_3	Y_3
Slow([x[,*]])	Y_1	Y_2	Y_2
SlowReverse(x[,*])	Y_1	Y_2	Y_2
Step([n[,*]])	Y_1		
StillOff([*])	Y_1		
Stop([*])	Y_1	Y_2	Y_2
TimePlay(h,m,s,f[,*])	Y_1	Y_2	Y_2
TitleGroupPlay(g[,*])	Y_1		
TitlePlay(t[,*])	Y_1	Y_2	Y_2
TrackPlay(g,t[,*])	Y_1		
VectorDraw(x,y,b[,*])	Y_1	Y_3	
VectorMove(x,y,b[,*])	Y_1	Y_3	
VideoBlending([a,c[,*]])	Y_1	Y_2	
Zoom([x,y[,*]])	Y_1	Y_2	

Recommended Commands and Special Effects Matchups

ITX Events

Events are integral to synchronizing DVD-Video with other media. With these events, web pages can be synchronized with the audio or video. For example, each ChapterEvent (start of new chapter) can change an HTML storyboard that corresponds to the movie. Time events can be used to coordinate advertising messages in HTML while the video is playing: when James Bond is driving his BMW, an appropriate web page (BMW or auto sales site) can automatically be displayed at the same time.

The value of events is that these external media do NOT have to be embedded or even be known at the time the DVD-Video is authored. This flexibility keeps DVD-Video authoring on schedule and greatly minimizes the authoring costs while adding valuable and unique features to each disc.

Events can be used by the calling application (HTML, C++, or other) to receive notification of DVD playback status. If a platform does not support an event, then an error code must be returned when its use is attempted. Supported events are:

ITX Events	**Description**	**CD-DA**	**Support Level**	**Range**
TitleEvent(t)	Called when title changes. Returns the new title number in t.		Base	1 - 99
ChapterEvent(c)	Called when chapter changes. Returns the new chapter number in c.		Base	1 - 99

TitleGroupEvent(g)	Called when title group changes. Returns the new title group number in g.		Base	1 - tbd
TrackEvent(t)	Called when the track changes. Returns the new track number in t.	Y	Base	1 - tbd
TimeEvent(e,t)	Called on time change. Returns elapsed time in e and total time in t. Both in milliseconds.		Base	e: 1 - 2^{31} t: 1 - 2^{31} (~2^{24} is practical limit
DisplayChange(x)	Called when slide/display list changes. Returns slide number in x.		Base	tbd
AngleEvent(x)	Called on angle change. Returns new angle number in x.		Base	1 - 9
StateEvent(x)	Called when play state changes (i.e., play to pause). Returns state in x. See CurrentState property for values.		Base	0 - 6
SpeedEvent(x)	Called when speed changes (i.e., play to scanning). Returns new speed in x.		Base	1 - 32

AngleEvent(c,m)	Called when user changes video angle. Returns current angle number in c and total number of angles in m		Base	c: 1 - 9 m: 1 - 9
MenuLanguageEvent(x)	Called when user changes menu language. Returns menu language number in x.		Base	1 - tbd
TextLanguageEvent(x)	Called when user changes text language. Returns text language number in x.		Base	1 – tbd
VideoErrorEvent(n)	Called when an error occurs. Returns error number in n.		Base	tbd
ParentalEvent(p,c)	Called when parental control changes. Returns level in p and country in c.		Adv	p: 1 – 8 c: tbd
KaraokeEvent(b)	Called when karaoke event changes. Returns 1 if karaoke track has begun playing, 0 if just finished.		Adv	0 – 1
EjectEvent()	Called when disc is ejected from device. No return value.	Y	Base	-
InsertEvent()	Called when disc is inserted into device. No return value.	Y	Base	-

ITX Events Summary

UOPSEvent(n)	Called when any UOP changes. Returns UOPs array in n.		Base	32 16-bit values
DomainEvent(x)	Called when domain changes. Returns domain in x.		Base	
MenuEvent(x)	Called when menu ID changes. Returns the ID of the new menu in x.		Adv	
MenuButtonEvent(x)	Called when user clicks a button on a menu. Returns the ID of the button selected in x.		Adv	1 - 32
MouseEvent(b,x,y)	Called when the user clicks either the left or right mouse button. Returns mouse button in b, x coordinate in x, and y coordinate in y.		Adv	b: tbd x: 0 - 720 y: 0– 480 or 576
AudioEvent(x)	Called when user changes audio track. Returns audio number in x.	Y	Base	1 - tbd
SubpictureEvent(x)	Called when user changes subpicture track. Returns subpicture number in x.		Base	1 - 32

ITX Properties

Properties can be used to find information about commonly used variables; such as time, title and chapter. All properties must be supported even if the advanced feature itself is not supported. Non-supported features may return a reasonable default value (for example if the zoom feature is not supported the zoom properties should always return 10000.) If a feature is not supportable and there is no reasonable value, then a -1 should be returned.

The following properties are supported:

ITX Property	**Description**	**CD DA**	**Support Level**	**Range**
CurrentElapsedTime	Elapsed time of current title (in milliseconds)	Y	Base	0 – 2^{31}
CurrentTotalTime	Total time of current title (in milliseconds)		Base	0 – 2^{31}
CurrentTitle	Currently playing title		Base	0 – 99
CurrentTitleGroup	Currently playing title group		Base	0 – 99
CurrentChapter	Currently playing chapter		Base	0 – 99
CurrentTrack	Currently playing track	Y	Base	0 – 99
CurrentDisplay	Currently playing display list item		Base	0 – 99
CurrentState	Current play state	Y	Base	0 – 6

	(0=None, 1=Scanning, 2=Stop, 3=Pause, 4=Play, 5=Slow Play, 6=Menu)			
CurrentDomain	Current domain		Base	tbd
CurrentAudio	Current audio track	Y	Base	0 - 99
CurrentSubpicture	Current sub picture track		Base	0 - 31
CurrentAngle	Current video angle		Base	1 - 9
CurrentMenuLang	Current menu language		Base	1 - 8
NumAudio	Number of audio languages/tracks currently available		Base	1 - 8
NumSubpicture	Number of subpictures currently available		Base	0 - 31
NumAngles	Number of angles currently available		Base	1 - 9
NumMenuLang	Number of menu languages available		Base	1 - 8
GetAudioLanguage(x)	Returns audio language (and extensions) for specified audio number x. Returned audio language is the 2-digit locale.		Base	0 - 99
GetSubpictureLanguage (x)	Returns subpiucture language (and extensions) for specified subpicture number x. Returned subpicture language is		Base	0 - 99

	the 2-digit locale.			
GetMenuLanguage(x)	Returns menu for specified menu number x. Returned menu language is the 2-digit locale.		Base	0 - 99
SupportedFeatures	Returns feature bits corresponding to capabilities of current system (Must be available before the navigator is in the play state). See section 4.1 and 4.2 for details.	Y	Base	32-bit mask
Version	Returns version of platform. This field can also be used to determine parsing of certain components in the SupportedFeatures property. Returns major version and minor version unique to each playback system.	Y	Base	Two 16-bit integers
CurrentZoomX	Current Zoom X value		Base	Unsigned 16-bit
CurrentZoomY	Current Zoom Y value		Base	Unsigned 16-

				bit
CurrentPanX	Current Pan X value		Base	Unsigned 16-bit
CurrentPanY	Current Pan Y value		Base	Unsigned 16-bit
CurrentMenuID	Current ID associated with currently selected menu		Base	0 - 99
NumLayers	Number of overlay layers currently possible (based on memory available at resolution 1, bpp=1) See note 8.	Y	Base	0 - 9
MaxLayers	Maximum number of simultaneous overlay layers supported See note 8.	Y	Base	0 - 9
MaxAlpha	Maximum number of alpha blending steps supported. (i.e. DVD subpictures requires 16 levels but hardware may support 256 levels).	Y	Base	16, 32, 64, 128, 256
MaxFast	Maximum number of fast speeds.		Base	0 - 99
MaxFastReverse	Maximum number of		Base	0 -

	reverse fast speeds.			99
MaxSlow	Maximum number of slow speeds. Could be zero if not supported.		Base	0 - 99
MaxSlowReverse	Maximum number of reverse slow speeds. Could be zero if not supported.		Base	0 - 99
MaxCmdQueue	Maximum size of the command queue	Y	Base	0 - 255
MaxBookmarks(x)	Maximum number of bookmarks based on x: 1: total in volatile memory 2: total in non-volatile memory 3: per disc in volatile memory 4: per disc in non-volatile memory	Y	Base	Unsigned 16-bit
NumBookmarks(x)	Number of bookmarks available based on x. (same as above)	Y	Base	Unsigned 16-bit
GetRelTime	Gets the relative time counter.	Y	Base	0 - 2^{31}
CurrentCmdQueue	Current number of empty slots in the command queue	Y	Base	0 - 255
GetDiscType	Gets the current disc type and sub-type. types: 0 = drive empty or	Y	Base	0 - 2^{16} low 8 bits

	unknown state 1 = DVD 2 = CD audio 3 = other 4 - 255 = reserved sub-types for DVD (bit fields): 0 = DVD-Video 1 = DVD-Audio 2 = DVD-ROM material present 3 = PCFriendly 4 = ITX 5-7 = reserved See section 2.1.2.4.1 for details			is an integer type; high 8 bits are bit fields
QueryNet	Gets Internet connection status 0 = not available, ever 1 = not currently avail 2 = available, not online 3 = online, speed unknown 4 = up to 28.8K 5 = up to 56K 6 = up to 128K 7 = up to 1.5M 8 = up to 10M 9 = >10M	Y	Base	0 - 9

ITX Properties Summary

Notes:

8. Layer Properties.

The MaxLayers property is how many simultaneous overlay layers the hardware can process or the software/hardware system can effectively emulate as simultaneous overlays in real time and blend with a full screen video. The NumLayers property returns the number of layers that can be created (but not necessarily used simultaneously) based on the amount of free memory currently available.

The concept of layer resolution is that a 720 x 480 image requires some number of bytes of data (depending on the bpp) at a resolution of 1. A resolution of 2 uses one data item for a 2 x 2 pixel area of the image (i.e. 4x less data). This allows a layer to be defined for markup that doesn't need high accuracy and/or a method for a platform to perform graceful degradation if not enough memory is available for a full resolution layer. Resolution 3 is a 3 x 3 pixel area, and is somewhat awkward. Resolution 4 is a 4 x 4 pixel area. No other resolutions are defined.

Disc Type Detection

The GetDiscType property requires that the type of disc in the player be available to the application. A disc may be only one of the following types:

0: drive empty or unknown state
1: DVD
2: CD
3: other

For a DVD disc, any number of the DVD sub-types may be detected and have their respective bits set as follows:

Bit number	Description	Detection method
0	DVD-Video	VIDEO_TS\VIDEO_TS.IFO file present
1	DVD-Audio	AUDIO_TS\AUDIO_TS.IFO file present
2	DVD-ROM material present	Any file in the main directory other than VIDEO_TS and AUDIO_TS directories
3	PCFriendly	DISC.ID file present
4	ITX	ITX.HTM file present
5-7	reserved	N/A

Disc Sub-Types bit fields

Browser Requirements

Web browsers and the software environment on each platform shall be capable of the following

Feature	Support Level
ITX features in para 2.1.2.2 - 2.1.2.4	Base/Adv

Presentation layer must properly interpret HTML with embedded video	Base
HTML version 4.0	Base
JavaScript version 1.2	Base
Platform determination (navigator.platform)	Base
Language determination (navigator.language)	Base
JavaScript handlers for	
Methods	Base
Properties	Base
Events	Base
Graphic support (JPG, GIF)	Base
Graphic support (BMP)	Adv
Animated GIF support	Adv
XML	Adv
Java support	Adv
Streaming media support	Adv
Macromedia Flash	Base/Adv
Macromedia Shockwave	Adv
QuickTime	Adv
Interfaces to common hardware features (ID, cookie, etc.)	Base

Browser Requirements Summary

HTTP Header Formatting (Base)

Each HTTP header should be formatted with the following information (in addition to standard HTTP header information:

- Language
- Screen resolution
- Hardware platform identifier and version
- Browser identifier and version

Cookies (Base)

Browser must be able to support cookie mechanism, which of course places a memory requirement on the hardware device. Cookie shall be placed by browser in local persistent memory and shall be readable only by a specific server and browser/hardware partner. Cookie shall contain:

- User/hardware ID: generated by computer software or by hardware platform (in case of set-top)
- Disc ID: generated by local hardware based on a hashing algorithm.
- BCA number: read from lead-in area of DVD

The following is matter of design choice.

- Format of cookie
- When to place cookie (i.e., insertion event)

Direct Connection to Navigator (Adv)

Ability to pass commands directly to DVD/CD navigator, such as:

- All DVD/CD navigation commands
- Additionally, must have ability to set GPRMs

Platform/Hardware Requirements

In order to provide a consistent baseline platform for ITX content developers it is important that the platform and hardware vendors properly support the ITX API. Not all hardware platforms will have identical capabilities. So it is important that each platform provide access to the features that are available and graceful degradation for those that are not supported - and provide this as feedback so that content developers understand how their content will function on different platforms.

Baseline Hardware Platform Requirements

Hardware platform vendors must provide hardware and interfaces capable of performing all the functions specified as base above to be ITX compatible. If the feature is not available it is important that it either be emulated or degrade gracefully in some manner. Items marked as advanced can be supported or not, but the Supported Features bits must accurately indicate what features are available.

It is expected that hardware platforms meet these minimum specifications:

- Support HTML 3.2 browser and other requirements in paragraphs 3.x
- Play video full screen down to a 4:1 downscale (180 x 120 (NTSC), 180 x 144 (PAL)).

bit	**ITX Baseline Command Group**	**Command list**
0	GRP_OPENVOB	Open(filename | type)

1	GRP_TRANSPORT	Play([*])
		Pause([*])
		Stop([*])
		FastForward([x[,*]])
		Rewind([x[,*]])
		NextChapter([*])
		PrevChapter([*])
		Resume([*])
		StillOff([*])

2	GRP_AUDIOTRANSPORT	NextTrack()
		PrevTrack()
		NextDisplay([*])
		PrevDisplay([*])

3	GRP_SEARCH	TitlePlay(t[,*])
		ChapterPlay(t,c[,*])
		TimePlay(h,m,s,f[,*])
		Menu(x[,*])

4	GRP_AUDIOSEARCH	TitleGroupPlay(g[,*])
		TrackPlay(g,t[,*])

5	GRP_UOP	UOPMask()

6	GRP_SELECT	UpButtonSelect([n])
		DownButtonSelect([n])
		LeftButtonSelect([n])
		RightButtonSelect([n])
		ButtonActivate()
		ButtonSelectAndActivate(n)

7	GRP_VFEATURES	SubPictureSelect (n)
		SubPictureEnable(n)
		AudioSelect (n)
		AngleSelect (n)
		MenuLanguageSelect(n)
		ParentalLevelSelect(n)
		ParentalCountrySelect(n)

		FullScreen(w[,*])
8	GRP_AFEATURES	TextLanguageSelect(n)

9	GRP_PC	Close()
		ShowControls(x,y[,*])
		ShowContextMenu()
		PopUpMenu()

10	GRP_DOWNSCALE	From HTML embedded object width and height parameters Zoom(x,y[,*]) (downscale required for baseline; upscale is advanced)
11-15	N/A	reserved (must return 0)

Baseline Capabilities Grouping

Advanced Hardware Platform Requirements:

Each advanced feature requires that it be fully supported for its feature bit to be enabled. However, different playback systems may have differing levels of support for some features, such as the number of bookmarks supported or the variety of special effects supported.

bit	**ITX Advanced Command Group**	**Command List**
16	GRP_FILEOPEN	Open(filename | type) Play files other

		than VOB and MPG. Audio: WAV MID Video: AVI

17	GRP_ADVPLAY	Slow([x[,*]])
		SlowReverse([x[,*]])
		Step([n[,*]])

18	GRP_HIDDEN	HiddenGroupPlay(g[,*])
		HiddenTrackPlay(g,t[,*])
		HiddenTimePlay(h,m,s[,*])

19	GRP_MENU	GotoMenuID(x[,*])

20	GRP_BOOKMARK	GotoBookMark(x[,*])
		SaveBookMark(x[,*])

21	GRP_MOUSE	AutoMouseHide(b)
22	GRP_KARAOKE	KaraokeSelect(x)

23	GRP_ZOOMPAN	Zoom([x,y[,*]])
		Pan([x,y[,*]])

24	GRP_BLEND	VideoBlending([a,c[,*]])

25	GRP_LAYER	CreateLayer(b,c,r,d,p)
		ChangePalette(b,p[,*])
		DestroyLayer(b)
		ShowLayer(b[,*])
		HideLayer(b[,*])

		DisplayImage(f,b,a[,*])

26	GRP_DRAW	SetVectorDraw(b,c,w[,*])
		SetVectorCorners(x1,y1,x2,y2)
		VectorMove(x,y,b[,*])
		VectorDraw(x,y,b[,*])

27	GRP_AUDIOMIX	SetMixVolume(x[,*])

28	GRP_QUEUE	FlushCmdQueue()
		SetRelTime([t])
		all optional timed command parameters and special effects

29	GRP_WEB	NetConnect([u[,*]])
30-31	N/A	reserved (must return 0)

Advanced Capabilities Grouping

Local Storage/Memory Requirements:

The only local storage requirement of ITX is minimal memory for the purpose of placing cookies. Optionally, the hardware platform can also support larger local memory for the purposes of caching web pages. More information: TBD.

Hardware Platform Considerations

- Some set-top players may not be able to access both DVD-Video and ROM content at the same time. The application will need to permit intelligent caching, and the platform will need to provide sufficient memory.

Directory Structure for Current PCFriendly Client:

```
/ROOT
  PCFRIEND.EXE (WIN)
  PCFRIEND (MAC)
  README (MAC)
  README (WIN)
  /COMMON
    /SETUP
      LANG.INI
      SETUP_EN.BMP
      SETUP_JA.BMP
      SETUP_FR.BMP
      LIC_EN.TXT
      LIC_JA.TXT
      LIC_FR.TXT
    /CONTENT
      general content (runs on multiple platforms)
  /MAC
    /SETUP
      PCFRIENDLY PLUG IN
      FLASH 4
    /CONTENT
  /WIN
    /SETUP
      PCFRIEND.ICO
      INUNINST.EXE
      UPDATE.DAT
    /CABINETS
      MAIN.CAB
      VIDEO.CAB
```

```
        OTHER.CAB
      /THIRDPTY
        /MACROMED
          SWFLASH.EXE
        /MSIE
          /EN
          /JA
```

User Operation Control:

	DVD-Video Only			DVD-Audio Only			DVD-Audio + Video		
Function	Menu	Title	Stop	AMGM	Title	Stop	AMGM	Title	Stop
TitlePlay Title_Group_Play	X	X	X		X	X	X	X	X
PTTPlay, TrackPlay	X	X	X		X	X	X	X	X
TimePlay	X	X	X		X	X	X	X	X
Stop	X	X			X		X	X	
TimeSearch		X			X			X	
PTTSearch, TrackSearch		X			X			X	
NextTrack					X			X	
PrevTrack					X			X	
NextPG	X	X					X		
PrevPG	X	X					X		
NextDisplay								X	
PrevDisplay								X	
ForwardScan	X	X			X		X	X	
BackwardScan	X	X			X		X	X	
Menu	X	X	X		X			X	X

Resume	X						X		
Up	X	X					X	X	
Down	X	X					X	X	
Left	X	X					X	X	
Right	X	X					X	X	
Enter	X	X					X	X	
ButtonSelectandActivate	X	X					X	X	
Pause	X	X			X		X	X	
MenuLanguageSelect			X						X
TextLanguageSelect						X			X
AudioChange	X	X	X		X	X	X	X	X
SubpictureChange	X	X	X					X	X
AngleChange	X	X	X					X	
ParentalLevel			X						
ParentalCountry			X						
VideoPresentationMode	X	X	X				X	X	X
KaraokeMode	X	X	X						
HiddenGroupPlay					X	X	X	X	X
HiddenTrackPlay					X	X	X	X	X
HiddenTimePlay					X	X	X	X	X

User Operation Control Summary

AMGM (Audio Manager Menu): Optional Visual Menu defined in the Audio Manager (AMG). The Audio Manager contains the information and data to control all Audio Title Sets (ATS), all Video Title Sets (VTSs) for Audio Titles and the AMGM.

Enumerations:

Item	Options	Value
Domain	First Play	1
	Video Manager Menu	2
	Video Title Set Menu	3
	Title	4
	Stop	5
	Unknown	-1
Menus	Title Menu	2
	Root Menu	3
	Subpicture Languages Menu	4
	Audio Languages Menu	5
	Angle Menu	6
	Chapter Menu	7
Play State	None	0
	Scanning	1
	Stop	2
	Pause	3
	Play	4
	Slow	5
	Step	6
	Unknown	-1
Speed State	Normal Speed	0
	Double Speed	1
	Slow Forward Speed	2
	Slow Backward Speed	3
	Fast Forward Speed	4

	Fast Backward Speed	5
	Step Speed	6
	Unknown	-1

Enumerations

With reference to Fig. **5**, a process **500** is described for providing an enhanced multimedia experience. In an operation **502**, DVD content is recorded onto a DVD disc. Then, in an operation, **504**, the HTML content is recorded onto the same disc. Thereafter, in an operation **506**, the disc is inserted into a client device. The client device can be, for example a personal computer having DVD capabilities and an Internet browser. The client device could also be a set top box. Then in an operation **508**, the DVD content is accessed by DVD software present on the client device. In a step **510**, the HTML data is accessed. The HTML content is preferably accessed by the browser software already present on the client device. The HTML content is can include data obtained via the Internet by the browser software under the direction of the HTML content recorded onto the disc. Also, the HTML content can consist of only the recorded HTML data with no need for Internet connection. Finally, in an operation **512**, the DVD content is supplemented with the HTML content to provide an enhanced multimedia event. The HTML content can be added to the DVD content in multiple ways. For example, the HTML content can be in the form of a picture within a picture, (e.g. a relatively small window within a DVD video). The HTML content could also be update data incorporated directly into a video or could be in the form of navigation commands or relevant Internet links.

With reference to Fig. **6**, a general process **600** is described for enhancing DVD-content with ROM content. In an operation **602**, DVD content is recorded onto a disc. The DVD content is in the form of standard DVD content familiar to those skilled in the art. Then, in an operation **604** DVD-ROM content is generated. This content is preferably HTML encoded content which can be read and operated on by standard Internet browsers. In an operation **606**, a plurality of directories are incorporated into the DVD-ROM content. The directories allow operation with multiple user device platforms. The directories preferably include common directories which can be used on several platforms sharing common properties as well as platform specific directories for use with platforms having unique interface requirements. Thereafter, in an operation **608**, the DVD-ROM content along with the directories is recorded onto the disc. Then, in an operation **610**, the user's particular device platform is determined. This operation occurs automatically upon the user's attempt to use the disc. Then, in an operation, **612**, a directory appropriate for use with the determined user device platform is selected from among the plurality of directories. This selected directory is called using Javascript function and appropriate tags associated with the directories.

While the invention herein disclosed has been described by means of specific embodiments and applications thereof, numerous modifications and variations could be made thereto by those skilled in the art without departing from the scope of the invention set forth in the claims.

CLAIMS

What is claimed is:

1. A method for providing enhanced content for play across multiple play platforms, comprising the steps of:

delivering media content to a client device;

delivering HTML content to a client device, the HTML content being accessible and usable by a plurality of client device platforms;

activating a browser to access the HTML content, the browser being located on and compatible for use with the client device;

activating firmware on the client device to access the media content; and

incorporating the accessed HTML content with the accessed media content.

2. A computer program for developing media content as recited in claim 1 further comprising a code segment that access the content recorded onto the recording medium by calling one of the plurality of directories, the directory being suitable for use with the platform of the client device.

3. A method for providing a common, cross platform framework for development of DVD-video content with DVD-ROM content as recited in claim 2 wherein the directories include HTML content.

4. A method for providing a common, cross platform framework for development of DVD-video content with DVD-ROM content as recited in claim 2 wherein the

directories contain JavaScript files.

5. A method for providing a common, cross platform framework for development of DVD-video content with DVD-ROM content as recited in claim 2 wherein the directories comply with ISO-9660 standards.

6. A method for providing a common, cross platform framework for development of DVD-video content with DVD-ROM content as recited in claim 2 wherein the directories contain platform specific code segments.

7. A method for providing a common, cross platform framework for development of DVD-video content with DVD-ROM content as recited in claim 2 wherein the directories support hybrid Windows/Macintosh discs, preserving resource forks for Macintosh operating systems.

8. A method for providing enhanced media content as recited in claim 2 wherein the HTML content is provided via a portable storage medium.

9. A method for providing enhanced media content as recited in claim 2 wherein the HTML content is provided via a network.

10. A method for providing enhanced media content as recited in claim 9 wherein the network is the Internet.

11. A method for providing enhanced media content as recited in claim 1 wherein the HTML content is overlaid onto the media content.

12. A method for providing enhanced media content as recited in claim 1 wherein the HTML content is in the form of textual script, which scrolls with the media content.

13. A method for providing enhanced media content as recited in claim 1 wherein the HTML scrolls synchronously with the media content and wherein selecting a portion of the script navigates the user to a corresponding location in the media content.

14. A method for providing enhanced media content as recited in claim 1 wherein the HTML content is in the form of an HTML page that starts a movie and checks for related Internet sites.

15. A method for providing enhanced media content as recited in claim 1 wherein the HTML content includes a page that links to a website.

16. A method for providing enhanced media content as recited in claim 1 wherein the HTML content includes a plurality of HTML files for accommodating a plurality of platforms of client devices.

17. A method for enhancing multimedia content, comprising the steps of:

providing a recording medium;

recording multimedia content onto the recording medium;

integrating HTML content with the multimedia content;

accessing the multimedia content and the HTML content, and

playing multimedia content and the HTML content having been accessed.

18. A method for enhancing multimedia content as recited in claim 17 further including the step of recording the HTML content onto the recording medium.

19. A method for enhancing multimedia content as recited in claim 17 wherein the multimedia content is DVD content accessed by DVD firmware on a client device, and where the HTML content is stored locally on the client device.

20. A method for enhancing multimedia content as recited in claim 17 wherein the multimedia content is DVD content accessed by DVD firmware on a client device and wherein the HTML content is provided from a remote server via a network.

21. A method for enhancing multimedia content as recited in claim 17, wherein:

the multimedia content is DVD content;

the HTML content is a textual script of the DVD content; and

selection of a portion of the textual script navigates the multimedia content to a corresponding location in the multimedia content.

22. A method for enhancing multimedia content as recited in claim 17 wherein the multimedia content is DVD content and wherein accessing the multimedia content activates the HTML content, linking the user to a server providing HTML content corresponding to the multimedia

content.

23. A method for enhancing multimedia content as recited in claim 17 wherein the video playback sends events that allow the HTML content to be synchronized.

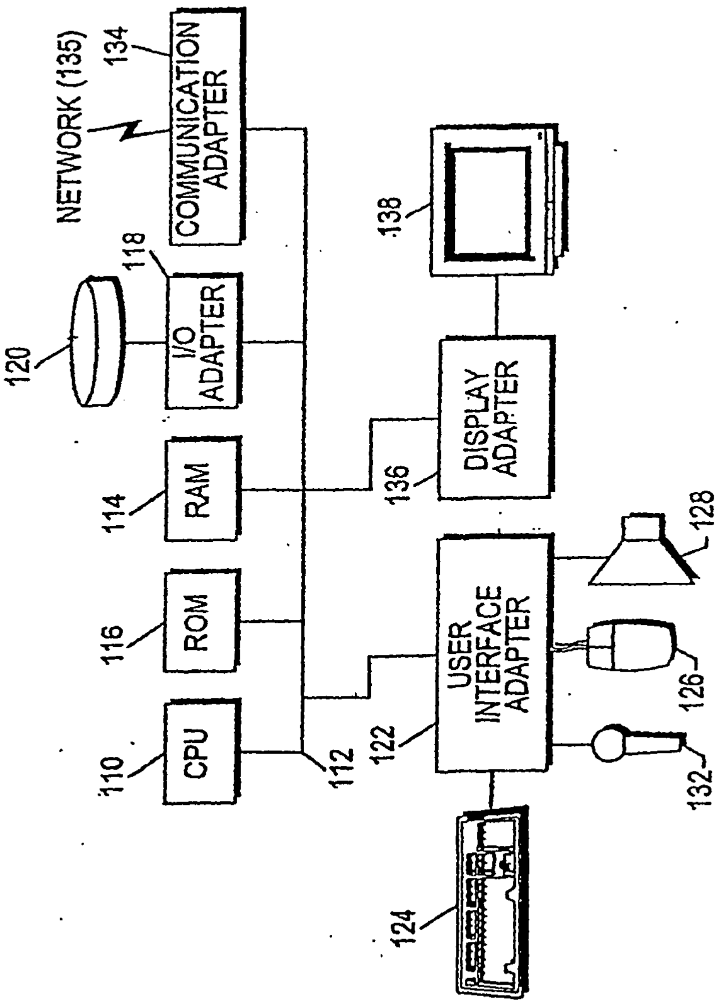

NETWORK (135)
134
COMMUNICATION ADAPTER
120
118
I/O ADAPTER
114
RAM
116
ROM
110
CPU
112
138
136
DISPLAY ADAPTER
122
USER INTERFACE ADAPTER
124
132
126
128

Figure 1

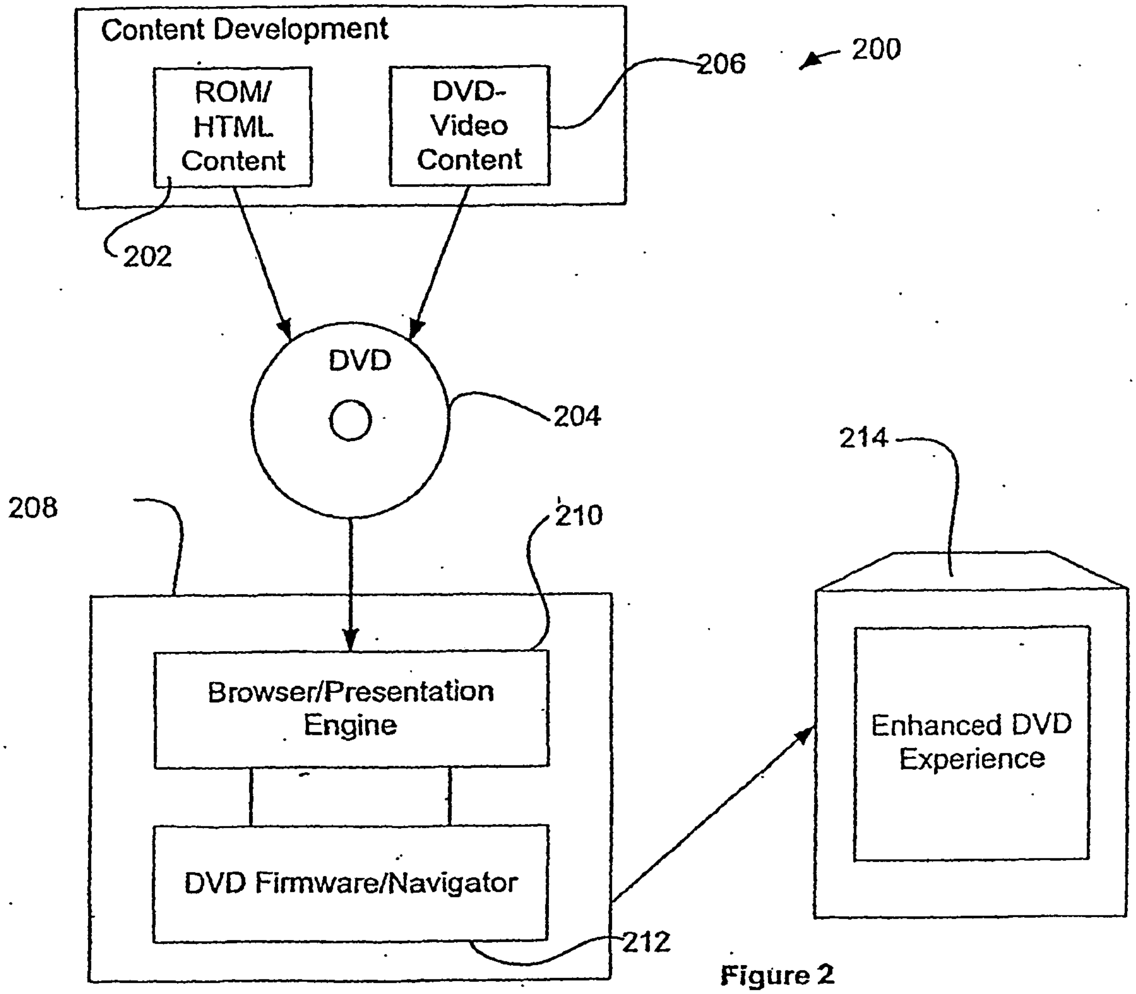

Content Development
ROM/ HTML Content
DVD- Video Content
206
200
202
DVD
204
214
208
210
Browser/Presentation Engine
DVD Firmware/Navigator
Enhanced DVD Experience
212

Figure 2

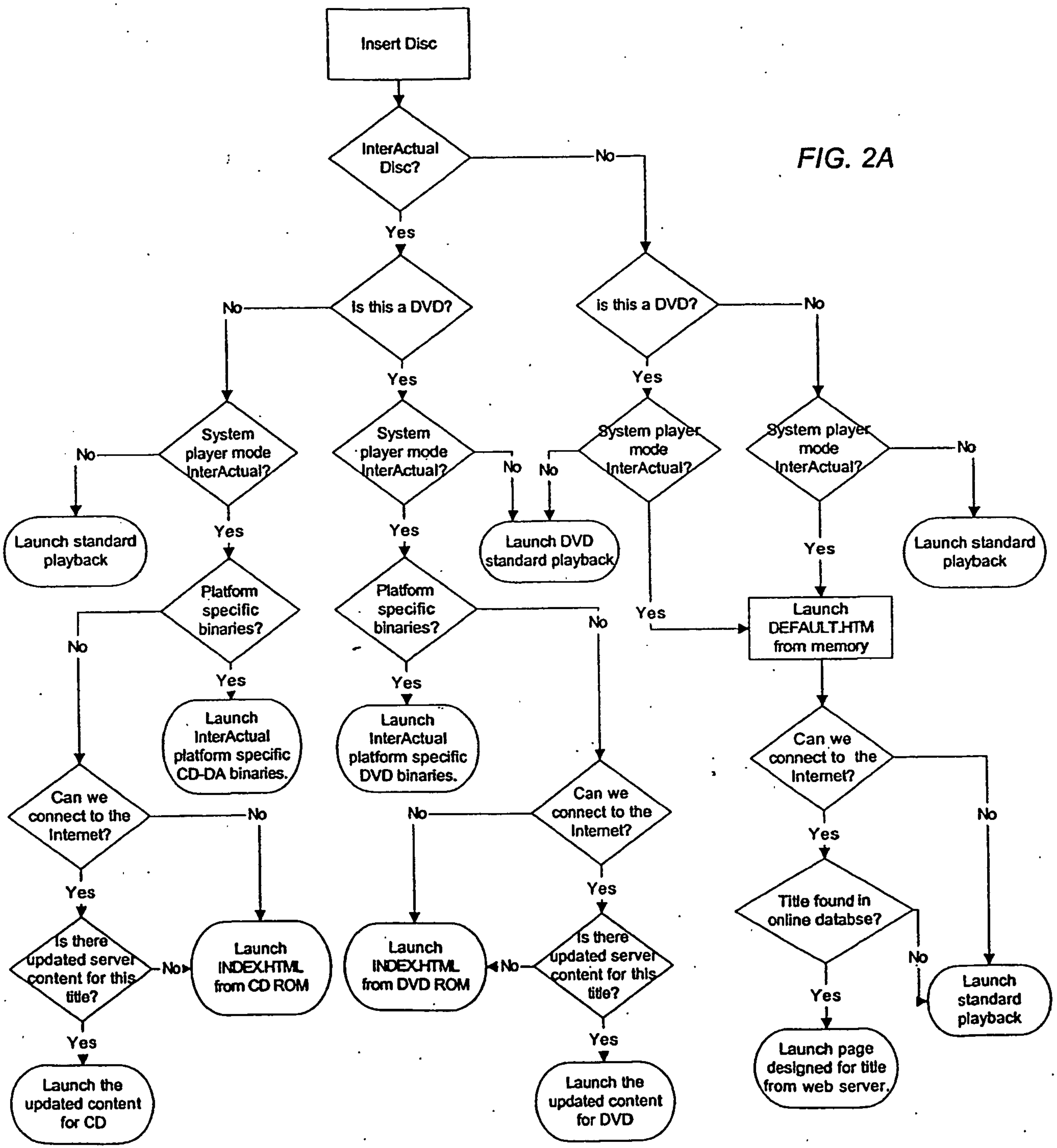

Insert Disc
InterActual Disc?
No
Yes
Is this a DVD?
No
Yes
Is this a DVD?
No
Yes
System player mode InterActual?
No
Yes
Launch standard playback
Platform specific binaries?
No
Yes
Launch InterActual platform specific CD-DA binaries.
Can we connect to the Internet?
No
Yes
Is there updated server content for this title?
No
Yes
Launch INDEX.HTML from CD ROM
Launch the updated content for CD
System player mode InterActual?
No
Yes
Launch DVD standard playback
Platform specific binaries?
No
Yes
Launch InterActual platform specific DVD binaries.
Can we connect to the Internet?
No
Yes
Launch INDEX.HTML from DVD ROM
Is there updated server content for this title?
No
Yes
Launch the updated content for DVD
System player mode InterActual?
No
Yes
System player mode InterActual?
No
Yes
Launch standard playback
Launch DEFAULT.HTM from memory
Can we connect to the Internet?
No
Yes
Title found in online databse?
No
Yes
Launch page designed for title from web server.
Launch standard playback

FIG. 2A

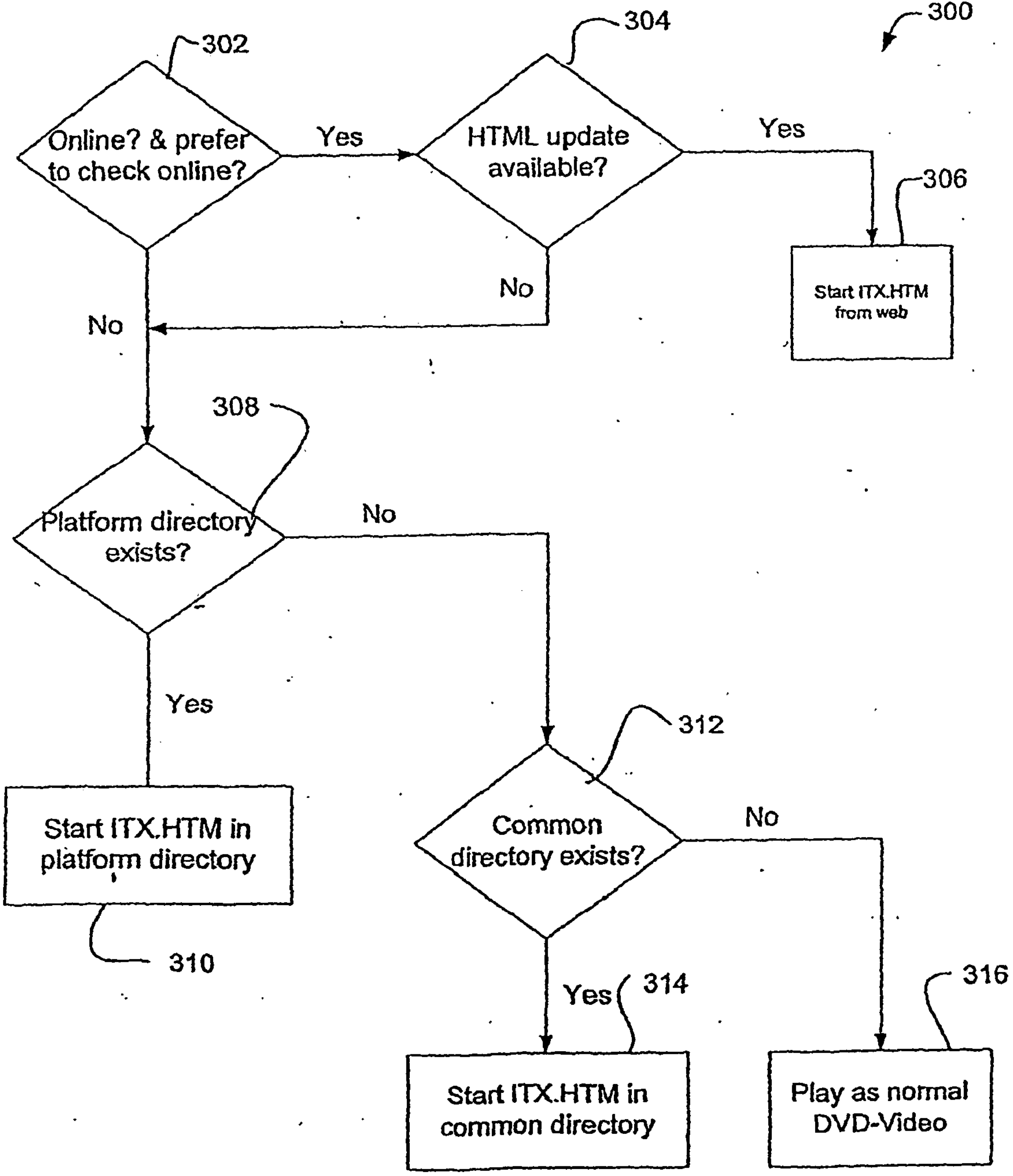

300
302
Online? & prefer to check online?
Yes
304
HTML update available?
Yes
306
Start ITX.HTM from web
No
No
308
Platform directory exists?
No
Yes
Start ITX.HTM in platform directory
310
312
Common directory exists?
No
Yes
314
316
Start ITX.HTM in common directory
Play as normal DVD-Video

Figure 3

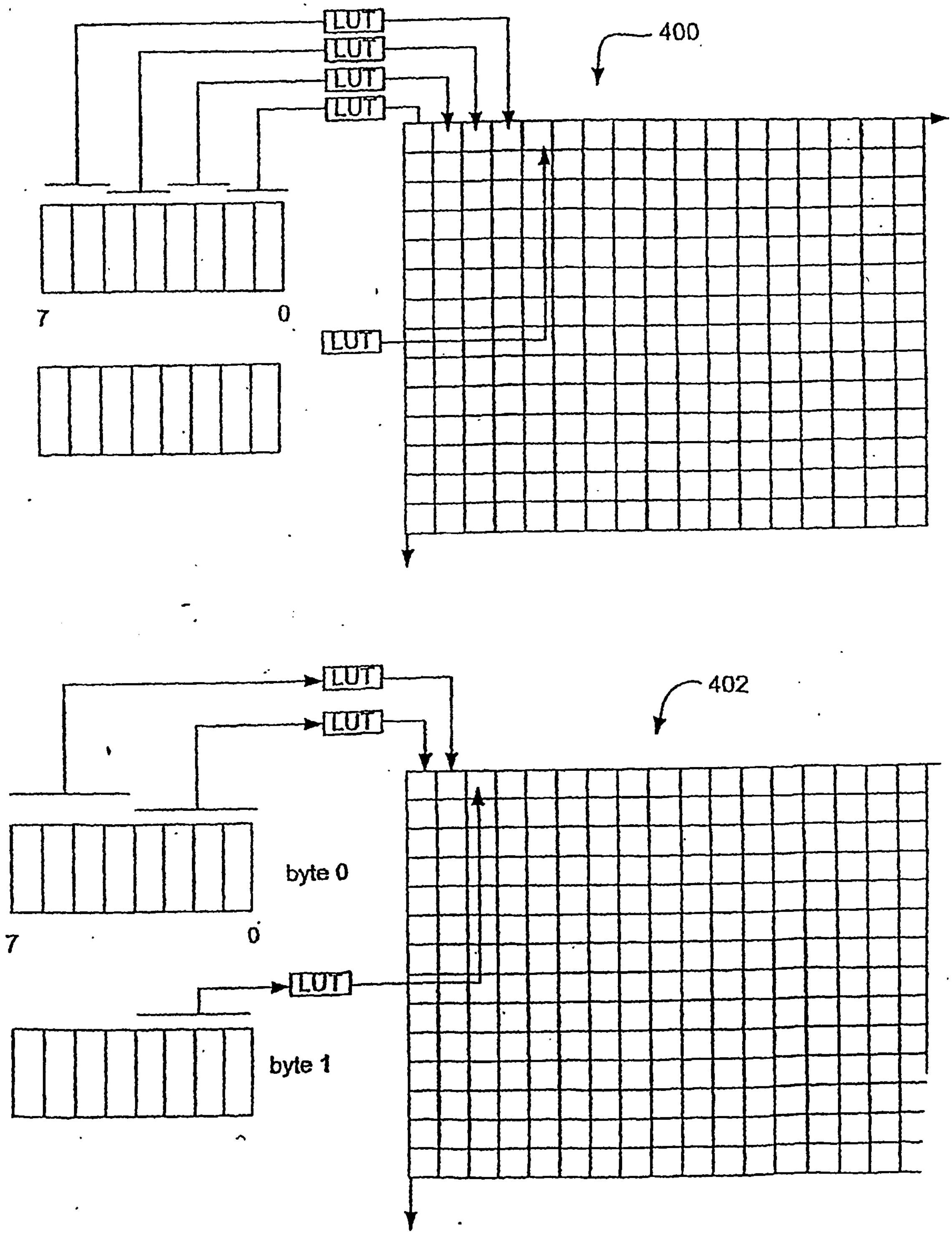


Figure 4

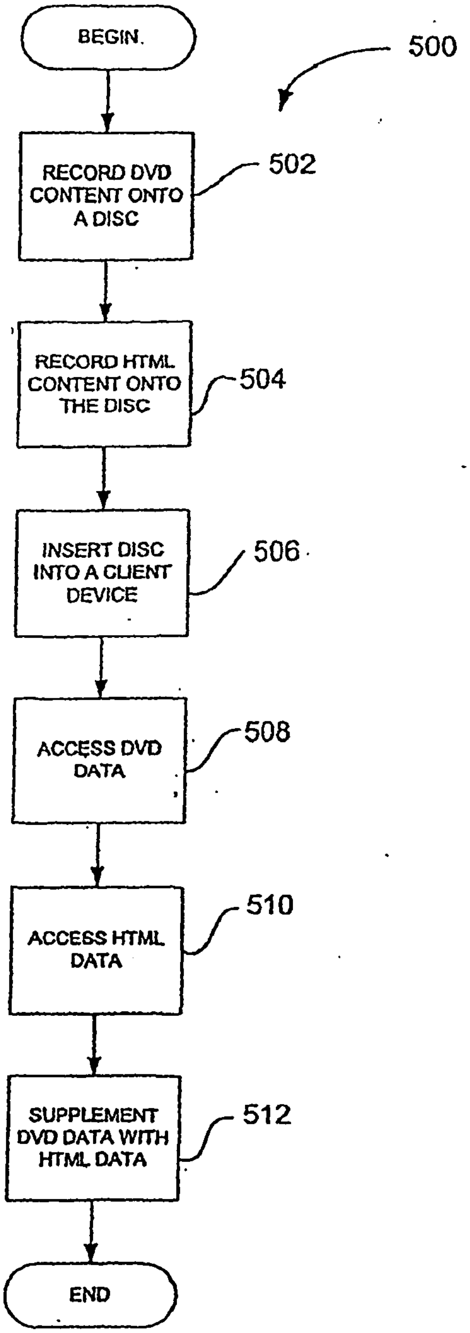

BEGIN
500
RECORD DVD CONTENT ONTO A DISC
502
RECORD HTML CONTENT ONTO THE DISC
504
INSERT DISC INTO A CLIENT DEVICE
506
ACCESS DVD DATA
508
ACCESS HTML DATA
510
SUPPLEMENT DVD DATA WITH HTML DATA
512
END

Figure 5

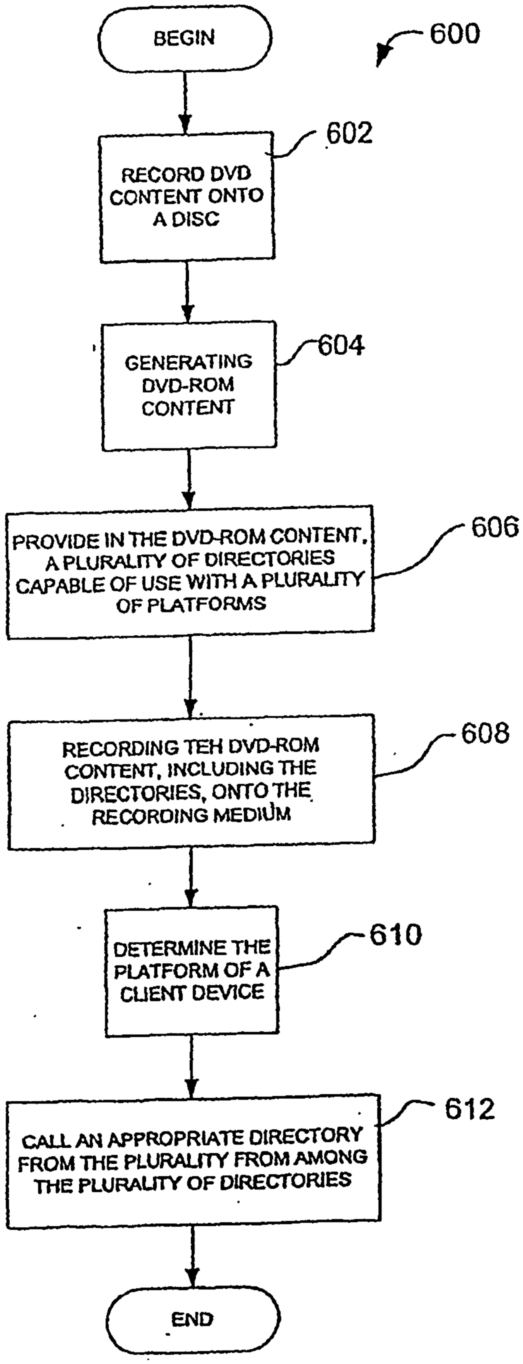

600
BEGIN
602
RECORD DVD CONTENT ONTO A DISC
604
GENERATING DVD-ROM CONTENT
606
PROVIDE IN THE DVD-ROM CONTENT, A PLURALITY OF DIRECTORIES CAPABLE OF USE WITH A PLURALITY OF PLATFORMS
608
RECORDING TEH DVD-ROM CONTENT, INCLUDING THE DIRECTORIES, ONTO THE RECORDING MEDIUM
610
DETERMINE THE PLATFORM OF A CLIENT DEVICE
612
CALL AN APPROPRIATE DIRECTORY FROM THE PLURALITY FROM AMONG THE PLURALITY OF DIRECTORIES
END

Figure 6

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/21187

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :G06F 15/16 ; G06F 17/00
US CL :709/246, 250; 369/34, 36; 345/150; 153; 707/513, 501
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 709/246, 250; 369/34, 36; 345/150; 153; 707/513, 501

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

West, East, PLus, IEEE
search term : updating, adding, modifying, enhancing multimedia content,,

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5,640,560 A (SMITH) 17 June 1997, col. 1 line 4 - col. 11 line 37.	1-23
A	US 5,943,304 A (KAMADA et al.) 24 August 1999, col. 1 line 9 - col. 32 line 67.	1-23
X,P	US 6,229,523 B1 (CZAKO) 08 May 2001, col. 1 line 6 - col. 13 line 7.	1-23
Y,P	US 6,230,174 B1 (BERGER et al.) 08 May 2001, col. 1 line 6 - col. 8 line 67.	1-23
Y,E	US 6,289,165 B1 (ABECASSIS) 11 September 2001, col. 1 line 10 - col. 4 line 24.	1-23

□ Further documents are listed in the continuation of Box C. □ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

24 SEPTEMBER 2001

Date of mailing of the international search report

21 NOV 2001

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
Peggy Harrod
FRANTZ B. JEAN
Telephone No. (703) 305-3900